ДЕПАРТАМЕНТ ОСВІТИ
ВІННИЦЬКОЇ МІСЬКОЇ РАДИ
КУ «МІСЬКИЙ МЕТОДИЧНИЙ КАБІНЕТ»
КОМУНАЛЬНИЙ ЗАКЛАД «ЗАГАЛЬНООСВІТНЯ ШКОЛА І-ІІІ СТУПЕНІВ №21
ВІННИЦЬКОЇ МІСЬКОЇ РАДИ»

«PYTHON 2.7 ДЛЯ ШКОЛЯРІВ»

номінація «Навчально–методичний посібник»

Любінецька Алла Павлівна
вчитель інформатики
«спеціаліст другої категорії»
тел. (097)644-23-85
Ткачук Юлія Павлівна
вчитель інформатики
«спеціаліст вищої категорії»
Тел. (097)405-82-89

.

м. Вінниця
2019

Автори-упорядники **Любінецька Алла Павлівна, Ткачук Юлія Павлівна**, вчителі інформатики комунального закладу «Загальноосвітня школа І-ІІІ ступенів №21 Вінницької міської ради»

Любінецька А.П., Ткачук Ю.П. Python 2.7 для школярів. Методичний посібник / А.П.Любінецька, Ю.П.Ткачук. – Вінниця: ММК, 2019. 121 с.

Рецензенти:

Кузьменко О.А., старший викладач, викладач інформатики та інформаційних технологій ДНЗ «Центр професійно-технічної освіти №1 м.Вінниці»

Вовк А.Р., заступник директора з навчально-виховної роботи комунального закладу «Загальноосвітня школа І-ІІІ ступенів №21 »

Рекомендовано методичною радою
комунального закладу «Загальноосвітня школа І-ІІІ ступенів №21
Вінницької міської ради»
(Протокол № 3 від 08.01.2019 р.)

В запропонованому посібнику подано теоретичний матеріал з алгоритмізації та основ програмування засобами середовища програмування Python 2.7, а також тести та різнорівневі завдання для закріплення знань та формування вмінь та навичок.

Матеріал подається невеликими блоками, що дає можливість зрозуміти та засвоїти навчальний матеріал.

Для вчителів інформатики та учнів 5-9 класів закладів загальної середньої освіти.

ЗМІСТ

ВСТУП

Мета базової загальної середньої освіти досягається шляхом реалізації таких завдань інформатичної освіти:

- визначати й формулювати у різноманітних життєвих ситуаціях задачі, для розв’язання яких можна залучити цифрові пристрої та інформаційні технології;
- знаходити, подавати, перетворювати, аналізувати, узагальнювати та систематизувати дані, необхідні для розв’язання життєвих задач;
- застосовувати алгоритмічний та системний підходи, створювати та аналізувати інформаційні моделі для ефективного розв’язання задач, що постають у житті, навчальній та професійній діяльності;
- вільно, відповідально й безпечно використовувати сучасні інформаційні технології та цифрові пристрої, а також самостійно опановувати нові;
- створювати інформаційні продукти, працюючи індивідуально або в команді;
- критично оцінювати інформацію та її вплив на людину і суспільство, переваги та ризики використання ІТ для себе, суспільства й довкілля;
- усвідомлювати етичні, суспільні, культурні та правові норми й дотримуватися їх під час роботи з інформацією та використання інформаційних технологій.

Однією з важливих умов міцності знань, умінь і навичок учнів є формування алгоритмічного мислення в процесі викладання навчальних предметів.

Інформатика стала базовим компонентом сучасної освіти, повноцінним загальнонауковим навчальним предметом. Вона відіграє дедалі більшу роль у житті суспільства, стає його важливим ресурсом. Аналіз змісту фахової діяльності людей масових професій і прогноз її розвитку дозволяють зробити висновок про зростання підготовки молоді в галузі інформаційних технологій.

Отже, школа повинна формувати теоретичну базу учнів з основ інформатики та практичні навички використання ними засобів інформаційних технологій у майбутній професійній діяльності.

Як відомо, для того, щоб комп’ютер міг виконувати потрібні нам задачі, ним повинна керувати програма.

Комп’ютерна програма – це набір інструкцій, які вказують машині послідовність дій для вирішення певної поставленої перед ним задачі. Для того, щоб комп’ютер зрозумів вказівки програміста, він повинен спілкуватися з ним однією мовою, яку називають мовою програмування.

На сьогоднішній день у світі існують сотні різних мов програмування.

Досвід програмістів підказує, що правильний вибір мови програмування є важливою частиною здійснення задуманого.

Мова програмування Python надає можливість “швидкого старту”. Написання перших програм потребує мінімальних знань синтаксису мови програмування. Ці знання поглиблюватимуться по мірі ускладнення навчальних завдань. Проста задача, як правило, матиме простий розв'язок на Python.

Python — це потужна мова програмування, якою легко оволодіти. Вона має ефективні структури даних високого рівня та простий, але ефективний підхід до об'єктно-орієнтованого програмування. Елегантний синтаксис Python, динамічна обробка типів, а також те, що це інтерпретована мова, роблять його ідеальним для написання скриптів та швидкої розробки прикладних програм у багатьох галузях на більшості платформ.

Інтерпретатор мови Python і багата стандартна бібліотека можуть бути отримані з сайту Python.org, і можуть вільно розповсюджуватися. Цей самий сайт має дистрибутиви та посилання на численні модулі, програми, утиліти та додаткову документацію.

Ми вирішили викладати програмування засобами середовища Python 2.7, починаючи з 5 класу, оскільки Scratch дітям в середній школі перестає бути цікавим. Тому виникла проблема щодо дидактичного матеріалу та відповідних підручників.

Опрацювавши навчальну програму з інформатики та різноманітні Інтернет-ресурси, ми дійшли висновку, що матеріалу з вивчення програмування засобами Python 2.7, адаптованого під звичайного школяра, практично немає. Тому виникла думка для своїх учнів самостійно створити посібник, у якому програмний матеріал

був би викладений доступно, послідовно, логічно, просто і цікаво.

Очікувані результати навчання вказано у змістовому розділі програми для кожної теми курсу в кожному класі.

Спираючись на пояснювальну записку до навчальної програми з інформатики для 5-9 класів, було змінено послідовність викладення окремих тем, не порушуючи змістових зв'язків між ними.

Орієнтовний план викладення матеріалу подано у наступній таблиці (Таблиця 1).

За програмою	*У посібнику*
5 клас	***5 клас***
Виконавці алгоритмів та їхні системи команд. Способи опису алгоритму. Програма. Середовище опису й виконання алгоритмів. Лінійні алгоритми. Алгоритми з розгалуженнями (перенесено в 6 клас). Алгоритми з повтореннями.	Встановлення Python. Працюємо в інтерактивному режимі. Середовище програмування для Python. Графічні можливості модуля Turtle. Команди переміщення черепашки. Створення найпростіших програм. Програми, що малюють коло. Команда циклу. Виконання команди. Малюнки з застосуванням циклів мовою Python.
6 клас	***6 клас***
Поняття про об'єкт у програмуванні. Властивості об'єкта. Створення програмних об'єктів Поняття події. Види подій. Програмне опрацювання події (перенесено на 7 клас).	Програмування та його види. Величини в програмуванні. Об'єкти та змінні. Введення та виведення величин. Розв'язування математичних задач. Алгоритми з повторенням для

Змінювання значень властивостей об’єкта в програмі. Вкладені алгоритмічні структури повторення та розгалуження. Розв’язання задачі методом поділу на підзадачі.	опрацювання величин. Алгоритми з розгалуженням для опрацювання величин. Модуль випадкових чисел. Відображення рисунків на координатній площині засобами мови програмування.
7 клас	***7 клас***
Величини. Змінні. Вказівка присвоювання (перенесено на 6 клас). Створення алгоритмів і програм з використанням змінних і різних алгоритмічних структур: лінійних, розгалужень і повторень. Опис моделей у середовищі програмування.	Типи даних у програмуванні. Опрацювання величин цілого типу. Опрацювання величин дійсного типу. Опрацювання величин рядкового типу. Створення вікна. Створення написів у вікні. Створення кнопок.
8 клас	***8 клас***
Сучасні мови програмування. Поняття об’єкта в мові програмування, його властивостей і методів. Графічний інтерфейс, основні компоненти програми з графічним інтерфейсом. Поняття елемента керування. Обробники подій, пов’язаних з елементами керування. Властивості та методи елементів керування.	Поняття класу та об’єкту. Пригадаймо підключення віконного інтерфейсу. Полотно. Екранні форми: список. Екранні форми: рамки. Екранні форми: перемикачі. Екранні форми: прапорці. Математичні рівняння і задачі.

Типи даних у програмуванні. Структура програми. Введення й виведення даних. Вирази. Логічні вирази та змінні й операції над ними. Умовні оператори (коротка та повна форма). Складені умови. Оператори циклу. Вкладені цикли. Пошук найбільшого та найменшого серед кількох значень.	
9 клас	***9 клас***
Поняття одновимірного масиву. Введення й виведення значень елементів масиву. Алгоритми опрацювання масивів: знаходження підсумкових величин, зокрема для елементів, що задовольняють задані умови, а також пошук у масиві за певними критеріями. Алгоритми впорядкування масиву. Поняття складності алгоритмів.	Поняття одновимірного масиву. Алгоритми опрацювання масиву. Знаходження елементів, що задовольняють задані умови. Знаходження підсумкових величин для елементів, що задовольняють задані умови. Алгоритми впорядкування масиву. Математичні рівняння і задачі.

Таб. 1

ОСНОВНА ЧАСТИНА

Посібник з програмування для 5 класу

§ 1. Встановлення Python

https://disted.edu.vn.ua/courses/learn/5383

Встановлення PYTHON в ОС Windows є простим.Досить мати інсталяцію. Завантажити можна з офіційного сайту **python.org.** Запустіть її на своїй машині; вкажіть папку, в яку інтерпретатор має бути встановлений.

Увага! Обирайте версію з врахуванням того, яку розрядність (32 чи 64 біта) має ваша версія Windows!

Наступний крок - вибір встановлюваних компонентів. Краще включити встановлення всього, що пропонується (зокрема, документацію). Повний комплект займає на жорсткому диску приблизно 20 мегабайт. Нарешті, потрібно задати назву групи для головного меню. Можна просто натиснути "Next". Програма покаже зведену інформацію про те, що і куди буде встановлено. Ще раз натискайте "Next" і чекайте закінчення. Натискайте "Finish" для виходу з програми встановлення.

Якщо у вас Windows 8 чи 10, то на робочому столі іконка може не з'явитися. Встановіть її, скопіювавши з меню "ПУСК".

§ 2. Працюємо в інтерактивному режимі

https://disted.edu.vn.ua/courses/learn/5383

Існує два способи використання інтерпретатора: командний режим і режим виконання програм з файлів. Якщо в командному рядку інтерпретатора Python ви наберете команду, то інтерпретатор одразу виведе результат її виконання. Такий режим називається **інтерактивним.**

ІНТЕРАКТИВНІСТЬ - від англ. ***interaction*** - обмін діями, взаємодія.

У широкому розумінні **для об'єктів інтерактивність** - це стан постійно перебувати у взаємозв'язку з іншими об'єктами, це залежність від різних станів інших об'єктів..

Ми будемо розглядати подібну взаємодію, але між користувачем (програмістом) та інтерпретатором мови Python:

- користувач пише команду, передає її на виконання;
- інтерпретатор намагається перекладасти введене, якщо спроба вдала, команда виконується і на екрані ми бачимо результат виконання, якщо спроба виявилась невдалою, на екрані відображаєься повідомлення про помилку;
- користувач бачить повідомленя на екрані, реагує на нього, змінюючи команду чи дописуючи нову;
- інтерпретатор знову починає роботу;
- далі реакція користувача, потім інтерпретатора ...

Отже, під **інтерактивністю** ми будемо розуміти здатність системи, без участі людини, активно і різноманітно реагувати на дії користувача.

У інтерактивному режимі запрошення введення наступної команди виводиться у вигляді первинного запрошення, зазвичай, три знаки "більше" (>>>); для продовження введення рядків видається вторинне запрошення (**...**).

У стандартний комплект постачання Python входить **інтегроване середовище розробки IDLE**, в якому редагувати програми буде набагато зручніше, ніж в простому текстовому редакторові. IDLE написане на Python. IDLE так само має вбудовану систему відлагодження (процес пошуку помилок).

У операційній системі Windows для виклику інтерпретатора досить в меню "**Пуск > Програми > Python 2.7.15 > IDLE**" викликати середовище розробки IDLE. Інтерпретатор друкує вітаюче повідомлення, констатуючи його номер версії і зауваження про авторське право, перед видачею першого запрошення:

```
Python 2.7.15 ( .....2:21:22) [MSC 32 bit (Intel)] on win32
Type "copyright", "credits" or "license" for more information.
IDLE 0.8 -- press F1 for help
>>> print "Hello world!"
Hello world!
```

Запрошення інтерпретатора >>> означає, що він готовий виконувати команди. Ми набрали команду **print "Hello world!"**, тобто дали вказівку вивести на екран рядок "Hello world!", і в наступному рядку інтерпретатор вивів те, що ми просили.

Є загальне правило - спочатку треба навчитися тому, як виходити з якоїсь ситуації, системи, програми, перед тим, як робити ще щось. Щоб завершити роботу з інтерпретатором Python, у Windows можна скористатися комбінацією клавіш **Ctrl+Z**.

Окрім чисел, Python також маніпулює з рядками (string). Інтерпретатор відрізняє рядок від числа за лапками, в які він поміщена.

```
>>> 'Hello world'
'Hello world'
>>>'2'  # тут двійка є текстом, а не цілим числом
'2'
```

Рядки можна зчіплювати один з одним за допомогою оператора +, тобто **конкатенувати**. Оператор * дозволяє отримати декілька копій рядка:

```
>>>'Help'+'A'
‘HelpA'
>>>'Help'*5
‘HelpHelpHelpHelpHelp'
```

Команда **print** також виводить значення вираження, але у випадку з рядками, вона виводить вміст рядка без лапок:

```
>>>print ‘Help'
Help
```

§ 3. Середовище програмування для Python

Для написання програм використовують **текстові редактори** або **інтегровані середовища розробки**, які включають в себе різні інструменти для роботи з кодом: засіб для написання коду (текстовий редактор), інтерактивний інтерпретатор, відлагоджувач тощо.

Текстові редактори та інтегровані середовища програмування для Python

IDLE — стандартний редактор Python. Встановлюється разом з Python для користувачів Windows, окремим пакетом для користувачів Linux.

Notepad++ - безкоштовний текстовий редактор вихідного коду, який підтримує велику кількість мов, в тому числі і Python. Лише для користувачів Windows.

PyScripter - інтегроване середовище розробки для мови програмування Python, працює під Windows. Поширюється безкоштовно.

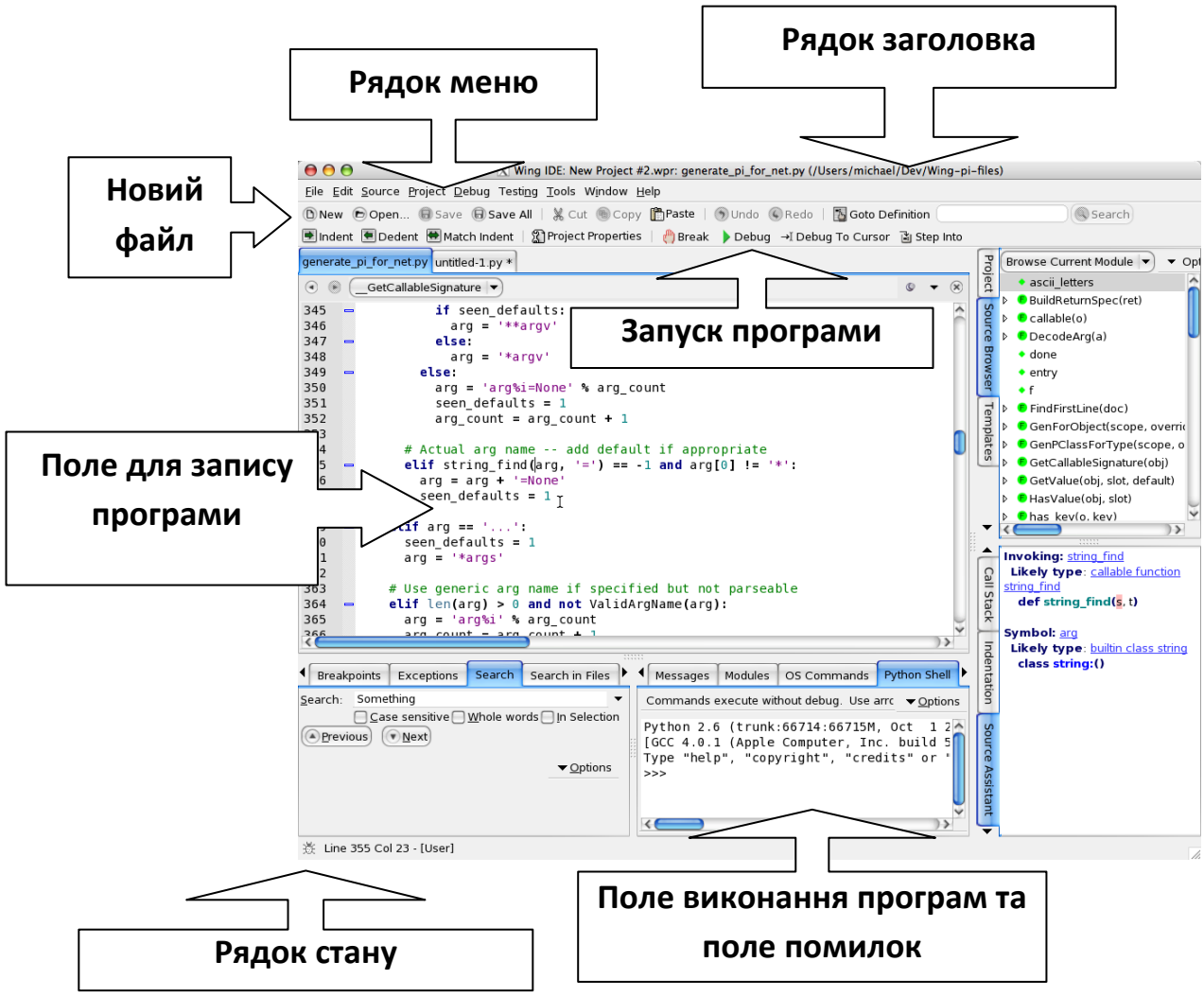
Wing IDE 101 - вільне інтегроване середовище для Python, розроблене для навчання програмістів-початківців. Для користувачів Linux, Windows і Mac OS X. Поширюється безкоштовно.

Geany - вільний текстовий редактор з базовими елементами інтегрованого середовища розробки, доступний для операційних систем Linux, Mac OS X і Windows.

Sublime Text 3 - кросплатформенний текстовий редактор вихідних текстів програм та інтегроване середовище розробки. Підтримує плагіни, розроблені за допомогою мови програмування Python. Sublime Text не є вільним чи відкритим програмним забезпеченням, але деякі його плагіни розповсюджуються з вільною ліцензією, розробляються і підтримуються спільнотою розробників. Проте, можна використовувати вільно, хоча часто з’являється повідомлення про придбання ліцензії. Для користувачів Windows, Mac OS X, Linux.

PyCharm — інтегроване середовище розробки для мови програмування Python. Підтримує веб-розробку на Django. PyCharm є власницьким програмним забезпеченням. Присутні безкоштовна версія **Community** з усіченим набором можливостей і безкоштовна версія **Edu** для навчальних закладів. PyCharm працює під операційними системами Windows, Mac OS X і Linux.

Інтерфейс середовища програмування



§ 4. Графічні можливості модуля Turtle

Основні алгоритмічні конструкції, зокрема цикли і багато іншого варто вивчати, використовуючи візуалізацію виконання програм. Такі можливості вперше були реалізовані в мові програмування LOGO. Це мова управління візком ("черепашкою") яка має хвіст, що може залишати за собою слід. Python має всі функції такої "черепахової" графіки, які зібрані в окремий модуль **turtle.**

Розглянемо, що таке модуль.

Модуль - це самостійна програма, що розширює функціонал вже існуючих програм.

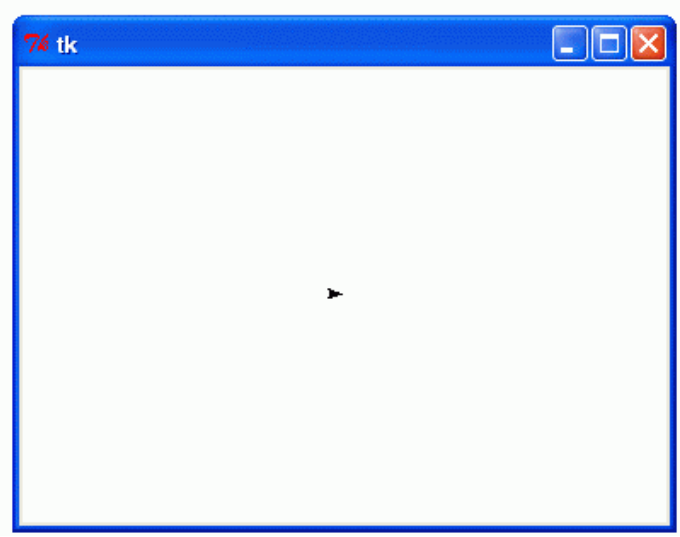
Функціонал - це перелік можливостей, що здатна виконувати програма.

Підключення модуля turtle у Python відбувається наступним чином:

from turtle import *

Ця стрічка розширює графічний функціонал.

Тепер ми можемо створювати різні картинки. Після запуску такого роду програм відкривається окреме вікно (з назвою **tk**) з вказівником, що має вигляд трикутника. Будемо називати його «**черепашкою**» (Мал.1).



Мал. 1

Черепашка дивиться строго **праворуч** і готова виконувати команди. Малює вона ніби пером. За замовчуванням на початку виконання перо в неї опущене і при пересуванні буде залишати **чорний** слід.

§ 5. Команди переміщення черепашки

forward(60) - проповзти вперед 60 кроків (пікселів).

left(50) - повернути наліво на 50 градусів.

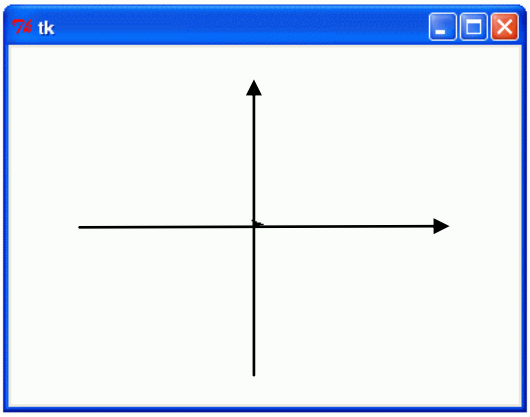
right(90) - повернути направо на 90 градусів.

circle(30) - намалювати коло радіуса 30, яке повністю знаходиться зліва від положення черепашки.

circle(-30) - намалювати коло радіуса 30, яке повністю знаходиться справа від положення черепашки.

Черепашка орієнтується у вікні як у системі координат. На початку виконання програми вона знаходиться у точці (0, 0) і готова перейти у будь-яку точку з

заданими координатами. Її система координат нічим не відрізняється від звичної для нас. Ця система координат невидима (Мал.2).



Мал. 2

goto(50, 90) - переміщення черепашки у координати (50, 90)

При цьому напрямок, в якому дивиться черепашка не змінюється, ця команда не повертає черепашку.

Команди малювання

down() - опустити перо. Після цієї команди черепашка почне залишати слід при будь-якому своєму пересуванні.

up() - підняти перо. Після цієї команди при пересуванні черепашка не буде залишати слід**.**

width(3) - встановити товщину сліду (пера) черепашки 3 пікселів.

color('red') - встановити червоний колір пера (сліду) черепашки. Колір має бути текстовим рядком з назвою кольору англійською мовою, наприклад, ‘red', ‘yellow', ‘green', ‘blue', ‘brown'.

fill() - використовується для малювання зафарбованих областей. Починаючи

begin_fill() – початок заливки,

end_fill() – кінець заливки.

Тестові завдання для перевірки знань

1. В який бік спрямована черепашка перед виконанням команд?
   a) Праворуч;
   b) Ліворуч;

c) Вгору;

d) Вниз.

2. Як називається графічний модуль?

a) Turtle;

b) Python;

c) Paint;

d) Forward.

3. Обрати команду, яка задає товщину лінії 5 пікселів.

a) forward(5);

b) width(5);

c) left(5);

d) right(5).

4. Обери правильний запис команди:

a) goto(50, 90);

b) goto(50);

c) width(-3);

d) circle(red).

5. Обрати команду, яка дозволяє черепашці повернути праворуч на 30.

a) forward(30);

b) left(30);

c) right(30);

d) circle(30).

6. Обрати команду, яка дозволяє черепашці при пересуванні не залишати слід.

a) down();

b) up ();

c) width();

d) fill().

7. Який слід залишає черепашка за замовчуванням?

a) білий;

b) чорний;

c) невидимий;

d) прозорий.

8. В записі **from turtle import** пропущено знак. Який?

a) ?

b) *

c) :

d) .

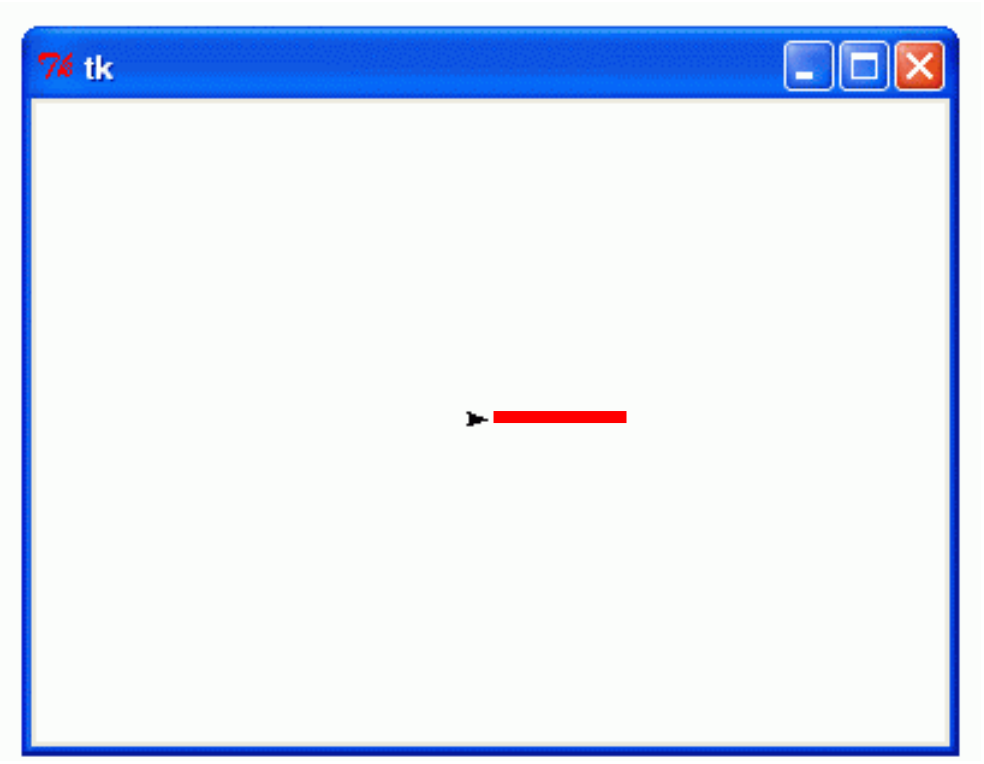
§ 6. Створення найпростіших програм

Проект 1.

Створимо програму, яка малює червону лінію довжиною 100 пікселів.

1. *from turtle import** підключення графічного модуля
2. *width(10)* товщина лінії 10 пікселів
3. *color('red')* колір лінії червоний
4. *forward(100)* довжина лінії 100 пікселів

Після запуску програми на виконання отримаємо таке зображення (Мал.3):



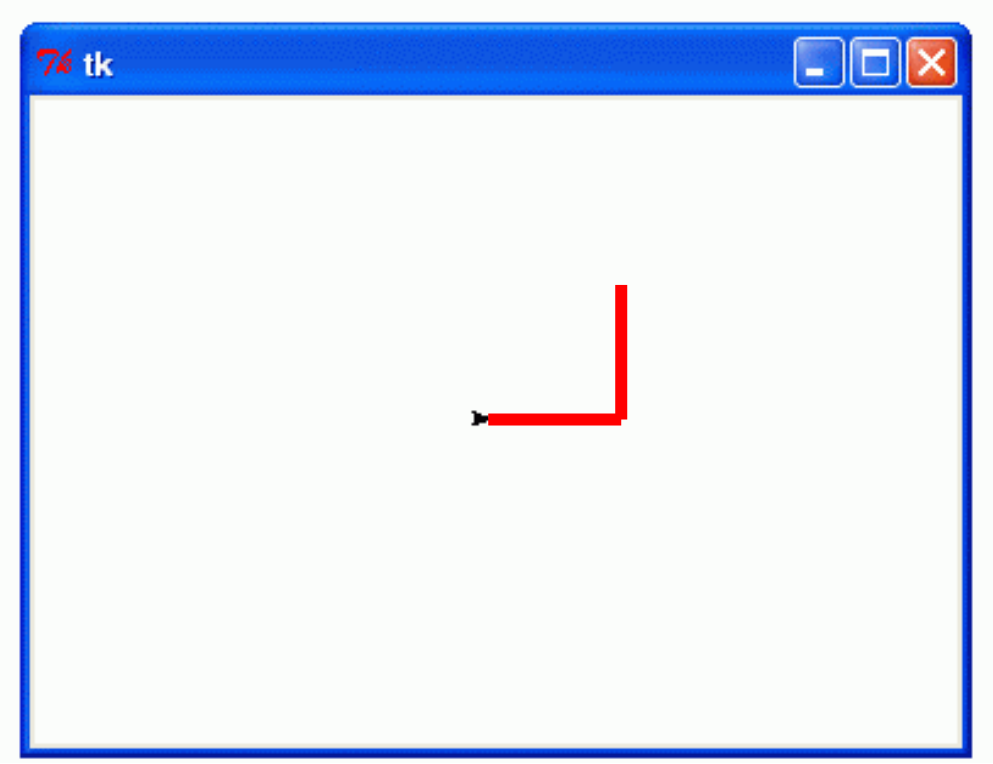
Мал.3

Проект 2.

Ускладнимо програму. Додамо елементи так, щоб утворився прямий кут.

1.	*from turtle import**	підключення графічного модуля
2.	*width(10)*	товщина лінії 10 пікселів
3.	*color('red')*	колір лінії червоний
4.	*forward(100)*	довжина лінії 100 пікселів
5.	*left (90)*	поворот ліворуч на 90 градусів
6.	*forward(100)*	довжина лінії 100 пікселів

Після запуску програми на виконання отримаємо таке зображення (Мал.4):



Мал.4

Завдання для самостійного виконання

І рівень

1. Написати програму, яка малює лінію синього кольору довжиною 200 пікселів і товщиною лінії 15 пікселів.
2. Написати програму, яка малює кут 60 градусів жовтого кольору довжиною лінії 100 пікселів і товщиною лінії 10 пікселів.

ІІ рівень

3. Написати програму, яка малює квадрат рожевого кольору зі стороною 150 пікселів і товщиною лінії 10 пікселів.

4. Написати програму, яка малює прямокутник зеленого кольору зі сторонами 150 і 80 пікселів і товщиною лінії 10 пікселів.

5. Написати програму, яка малює квадрат зі стороною 200 пікселів і товщиною лінії 10 пікселів, кожна сторона якого має інший колір.

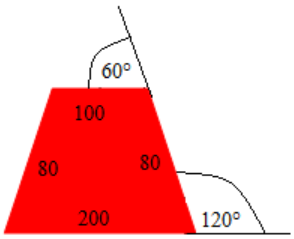
6. Написати програму, яка малює зафарбований квадрат оранжевого кольору зі стороною 100 пікселів і товщиною лінії 5 пікселів.

III рівень

7. Написати програму, яка малює трикутник фіолетового кольору зі стороною 150 пікселів і товщиною лінії 10 пікселів.

8. Написати програму, яка малює ромб рожевого кольору зі стороною 150 пікселів і товщиною лінії 10 пікселів.

9. Написати програму, яка малює трапецію синього кольору зі сторонами, вказаними на малюнку 5.



Мал.5

Завдання для розумників

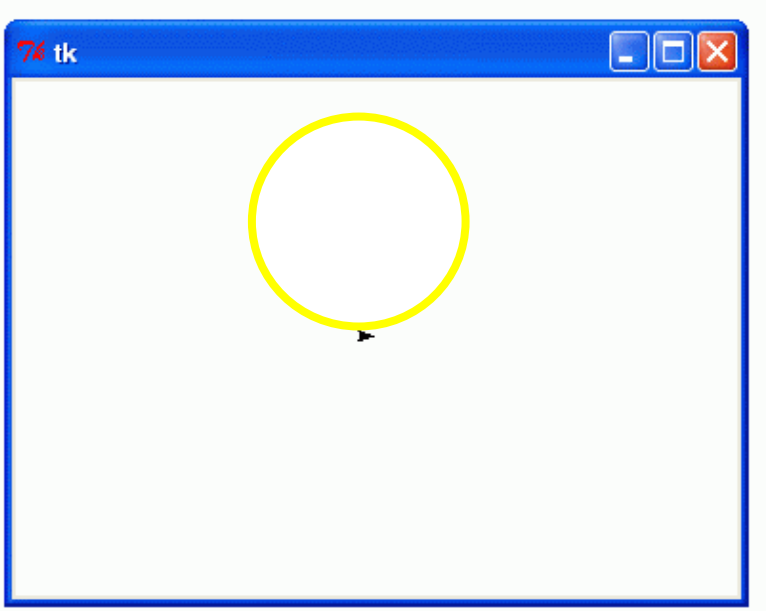
10. Створити власний проект, використовуючи вивчений матеріал.

§ 7. Програми, що малюють коло

Напишемо програму, яка малює жовте коло радіуса 100 і товщиною лінії 10

1. ***from turtle import **** підключити модуль turtle
2. ***color('yellow')*** встановити колір сліду
3. ***width(10)*** встановити товщину лінії
4. ***circle(100)*** намалювати коло

Результат виконання програми (Мал.6):



Мал.6

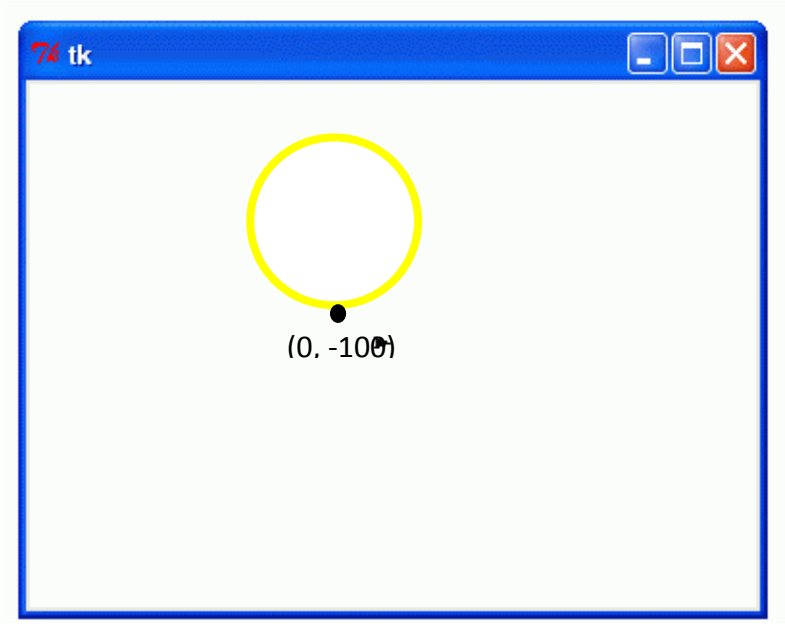
Залежно від того, куди дивиться черепашка, отримуємо різні положення кола.

Для малювання кола в різних місцях екрану використовуємо команду **goto(x,y).**

Зверніть увагу, що спочатку ми переносимо черепашку, а потім малюємо з нової позиції коло, а не навпаки.

1. ***from turtle import ****
2. ***up()*** піднімаємо перо
3. ***goto(0,-100)*** переміщуємо черепашку в точку з координатами (0,-100)
4. ***down()*** піднімаємо перо
5. ***color('yellow')***
6. ***width(10)***
7. ***circle(100)***

Результат виконання програми (Мал.7):



Мал.7

Завдання для самостійного виконання

І рівень

1. Написати програму, яка малює коло червоного кольору радіусом 50 пікселів**.**

2. Написати програму, яка малює коло червоного кольору радіусом 50 пікселів з точки (-50,-50).

ІІ рівень

3. Написати програму, яка малює два кола однакового кольору та однакових радіусів, що перетинаються**.**

4. Написати програму, яка малює два кола однакового кольору та різних радіусів, що перетинаються**.**

5. Написати програму, яка малює два кола різного кольору та різних радіусів, що перетинаються**.**

6. Написати програму, яка малює два кола однакового кольору та однакових радіусів, що не перетинаються**.**

7. Написати програму, яка малює два кола однакового кольору та різних радіусів, що не перетинаються**.**

8. Написати програму, яка малює два кола різного кольору та різних радіусів, що не перетинаються**.**

9. Написати програму, яка малює зафарбоване коло довільного радіуса.

III рівень

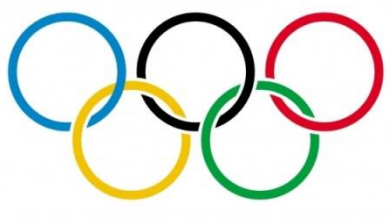
10. Написати програму, що малює квітку, яка складається з 4 кіл одного кольору.

11. Написати програму, що малює квітку, яка складається з 6 кіл одного кольору.

12. Написати програму, що малює квітку, яка складається з 4 кіл різних кольорів.

IV рівень

13. Написати програму, яка малює олімпійські кільця.

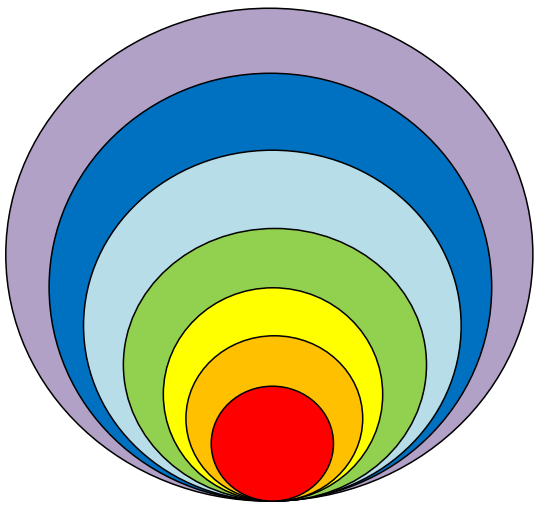


14. Написати програму, яка створює довільний малюнок, що складається з геометричних фігур.

15. Написати програму, яка малює «Хвіст павича».

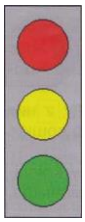
Параметри:

1. *Коло фіолетове, радіус 240.*
2. *Коло синє, радіус 210.*
3. *Коло блакитне, радіус 180.*
4. *Коло зелене, радіус 150.*
5. *Коло жовте, радіус 120.*
6. *Коло оранжеве, радіус 90.*



7. Коло червоне, радіус 60

16. Написати програму, яка малює світлофор.



17. Написати програму, яка малює гусеницю.

§ 8. Команда циклу. Виконання команди

https://disted.edu.vn.ua/courses/learn/5431

Більшість алгоритмів містять серії багатократно повторюваних дій. Для їх опису використовується команда циклу.

Алгоритми (програми), що містять команду циклу, називаються **циклічними.**

Кожна циклічна команда має тіло циклу - деяку серію команд, яку інтерпретатор повторює.

Крім того, кожна циклічна команда має умову виконання повторень. Що таке умова?

Умова - це вислів, про який можна сказати виконується він чи ні.

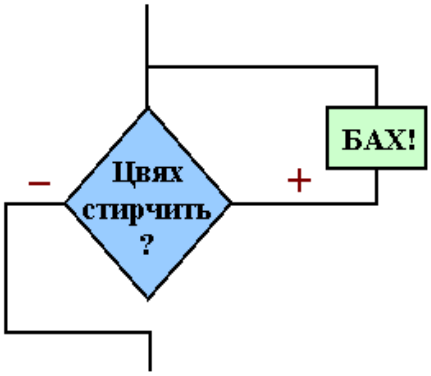
Якщо умова виконується, то говорять, що вислів **істинний**. Якщо ж умова не виконується, то вислів є **хибним.**

Які з наступних висловів істинні, а які хибні?

1. Всі горобці мають дзьоб
2. Деякі люди побували на Північному полюсі

3. В жодному місяці немає 50 днів
4. Всі дерева ростуть у лісі
5. Всі діти вчаться у школі
6. Дехто з учнів нашого класу їздив у Київ
7. Деякі учні нашого класу побували на Марсі
8. Всі учні нашого класу живуть на одній вулиці
9. Жодне слово не починається з букви "м"
10. Всі слова починаються з букви "а"
11. В кінці деяких речень стоїть знак питання
12. Доба має 24 години

Приклад 1. Нам необхідно забити цвях у дошку. Що ми робимо? Ми беремо молоток і вдаряємо по шляпці, якщо цвях все ще стирчить над поверхнею, то ми знову вдаряємо по шляпці. Так продовжується, доки умова істинна. Як тільки умова Намалюємо графічно, як це може виглядати (Мал.8).



Мал.8

Умовою тут виступає запитання: «Цвях стирчить?», тілом циклу є команда «Бах!», що символізує удар по шляпці цвяха.

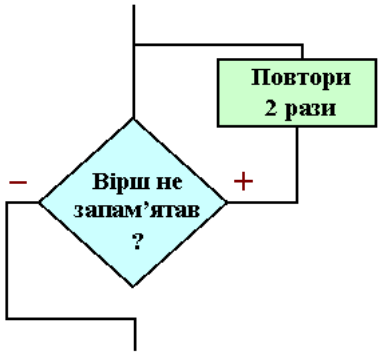
Знаком «+» позначається виконання умови, цим знаком починається гілка з тілом циклу.

Знаком «-» позначається невиконання умови, цим знаком починається гілка виходу з циклу.

Приклад 2. Як вивчити вірш?

У цьому прикладі умовою є «Вірш не запам'ятав?», а тілом циклу - «Повтори 2 рази» (Мал.9).

Розглянемо, який загальний вигляд команди циклу на алгоритмічній мові та мові Python:



Мал.9

алгоритмічна мова	мова Python
поки умова	**while умова :**
пц	**тіло циклу**
тело циклу	
кц	

(пц - **п***очаток* **ц***иклу,* кц - **к***інець* **ц***иклу***)**

Зверніть увагу на двокрапку після умови. Це невід'ємна частина команди циклу на мові Python.

Серія команд, що повторюються, на Python відділяється **відступами.**

Цей оператор виконується наступним чином. Перевіряється умова. Якщо вона виконується, тобто має значення TRUE (правда, істина, +), то виконується тіло циклу. Знову перевіряється умова, якщо вона істинна, то знову виконується тіло циклу. Так повторюється, поки умова не перестане виконуватися. Тобто прийме значення FALSE (неправда, хиба, -). Як тільки це трапилося повторення дій припиняється, управління передається оператору, що є наступним за оператором циклу.

Отже, інтерпретатор повторює тіло циклу, поки умова циклу залишається істинною.

! Якщо умова не виконується при першій перевірці, то тіло циклу не виконується жодного разу.

Очевидно, що один з операторів, що знаходяться в тілі циклу, повинен впливати на умову, інакше цикл буде виконуватись вічно.

§ 9. Малюнки з застосуванням циклів мовою Python

https://disted.edu.vn.ua/courses/learn/5432

Напишемо програму, яка малює квадрат стороною 60, починаючи з правого верхнього кута з координатами (-20, 30).

Проаналізуємо програму:

```
from turtle import *
up()
goto(-20,30)
down()
left(180)
forward(60)
left(90)
right(180)
```

4 рази

Використаємо для написання програми команду циклу, що виконується 4 рази. Тілом цього циклу команди, що треба повторити. Матимемо:

```
from turtle import *
up()
goto(-20,30)
down()
left(180)
i=1
while i<=4:
    forward(60)
    left(90)
    i=i+1
```

Увага! Тіло циклу відділено відступами!

Напишемо програму, яка малює коло, з використанням циклу.

```
from turtle import *
color('yellow')
width(10)
i=1
while i<=360:
    forward(1)
```

```
left(1)
i=i+1
```

Завдання для самостійного виконання

І рівень

1. Скласти програму, яка малює зафарбоване коло, використовуючи команду циклу.

ІІ рівень

2. Скласти програму, яка малює квітку, що складається з 4 кіл.

3. Скласти програму, яка малює квітку, що складається з 6 кіл.

4. Скласти програму, яка малює квітку, що складається з 6 відрізків.

5. Скласти програму, яка малює квітку, що складається з 9 відрізків.

6. Скласти програму, яка малює квітку, що складається з 12 відрізків.

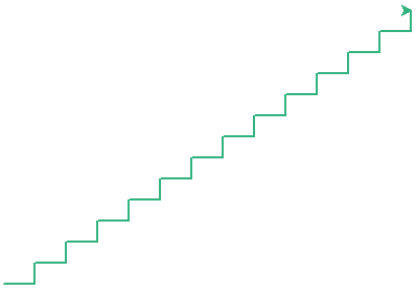
ІІІ рівень

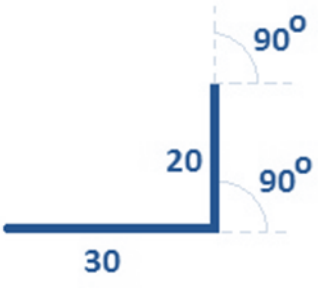
7. Скласти програму, яка малює квітку з 6 квадратів.

8. Скласти програму, яка малює квітку з 9 квадратів.

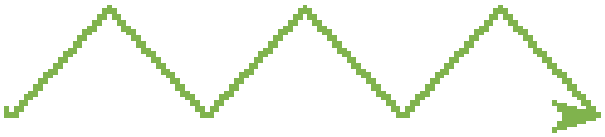
9. Скласти програму, яка малює квітку з 12 квадратів.

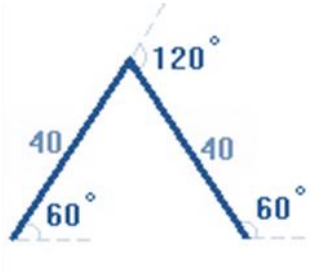
10. Написати програму, яка зображуватиме сходинки, починаючи з точки (-190, -120).



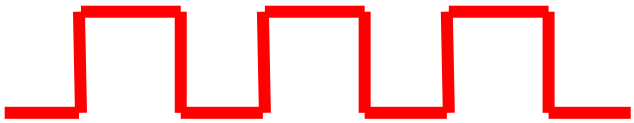


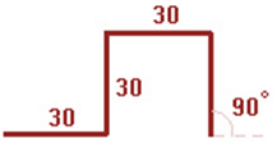
11. Написати програму, яка зображуватиме пружинку, починаючи з точки (0, -120).



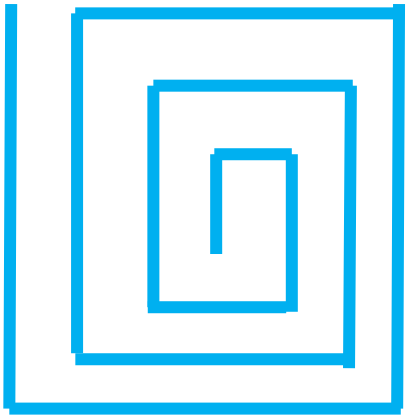


12. Написати програму, яка зображуватиме візерунок, починаючи з точки (0, -100).





13. Написати програму, яка малює лабіринт.



14. Виконайте довільний малюнок із застосуванням циклу.

15. Скласти програму, у якій черепашка прописує ваше ім’я.

Проекти

16. Скласти програму, яка малює ракету.

17. Скласти програму, яка малює зоряне небо та місяць.

18. Скласти програму, яка малює садок з деревами та кущами.

19. Скласти програму, яка малює машину на дорозі.

20. Скласти програму, яка малює акваріум з рибками.

21. Скласти програму, яка малює Сонячну систему.

22. Скласти програму, яка малює теплохід на морі.

23. Скласти програму, яка малює герб України.

24. Скласти програму, яка малює прапор України.

25. Скласти програму, яка малює квіткову галявину.

Посібник з програмування для 6 класу

§ 1. Програмування та його види

Програма визначається як “алгоритм, тобто послідовність інструкцій, який записано мовою, що зрозуміла комп’ютерові. Програма складається з підсистем, які певним чином взаємодіють між собою.

Програма являє собою певну модель, яка описує з деяким наближенням закономірності реального світу.

Програма є сукупністю правил, які дозволяють отримувати різноманітні наслідки (вихідні дані) з відомих передумов (вхідні дані).

Програма є засобом для вирішення конкретних практичних задач.

Відповідно до цього можна підходити до **програмування** як до процесу конструювання програм з різних точок зору.

Виділяють чотири основні типи програмування:

1. **Процедурне програмування.**

Концепція процедурного програмування є історично першою та найбільш близькою до класичного визначення програми.

В основі програми, побудованої за процедурними принципами, лежить послідовна зміна вхідних даних, поки не буде отриманий результат, причому кожна операція розписується в явному вигляді. Дані, з якими оперує програма, зберігаються в іменованих ділянках оперативної пам’яті, які називаються змінними (Паскаль і Сі).

2. **Функціональне програмування.**

Виконання програми розглядається як виклик деякої функції, яка, в свою чергу, може викликати інші функції.

3. **Логічне програмування.**

В основі виконання програми лежить механізм виведення нових фактів з даних фактів згідно із заданими логічними правилами. Найбільш відомий представник – Пролог.

4**. Об’єктно-орієнтоване програмування** – найбільш стрімко розвивається в

сучасний час.

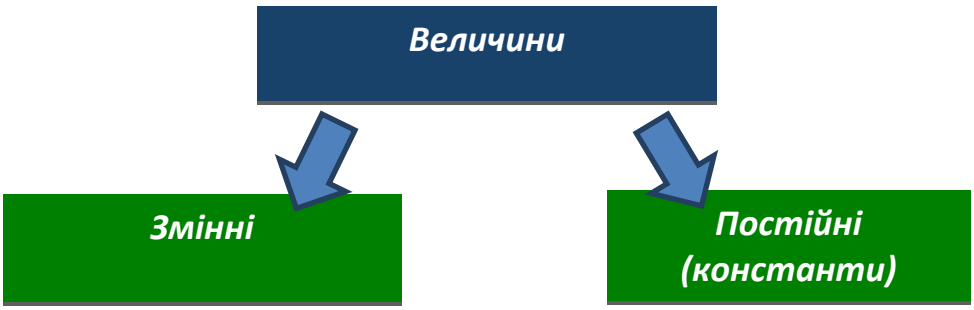
Об’єктна програма розглядається як сукупність паралельно існуючих об’єктів, які взаємодіють між собою. Кожний об’єкт вміє виконувати певні операції та характеризується певною поведінкою

Принципи об’єктно-орієнтованого програмування:

1. Все є об’єктами.
2. Всі дії та розрахунки виконуються шляхом взаємодії між об’єктами, при якій один об’єкт потребує, щоб інший об’єкт виконав деяку дію.
3. Об’єкти взаємодіють, надсилаючи і отримуючи повідомлення – запити на виконання дії.
4. Кожен об’єкт має незалежну пам’ять, яка складається з інших об’єктів.
5. Кожен об’єкт є представником (екземпляром, примірником) класу, який виражає загальні властивості об’єктів.

§ 2. Величини в програмуванні

Для опису об'єктів і процесів у матеріальному світі ми використовуємо величини. З прикладами величин ви стикаєтеся щодня: відстань між будинком і школою, температура повітря тощо. Кожна величина характеризується певним значенням та одиницями, в яких вимірюється це значення. Величина має ім'я та може набувати різних значень із деякої множини допустимих значень. Тип цих значень визначає тип самої величини. Для посилань на величини у виразах під час створення програми використовують імена величин. Позначення імен називають також ідентифікаторами. Ідентифікатори добирають у вигляді деякого скінченного впорядкованого набору літер і цифр, який починається з літери або символу підкреслення _.



Величина, яка має одне й те саме значення в будь-які моменти часу, називається **постійною**, або константою. Константам присвоюються значення в описовій частині програми і в процесі виконання програми їх змінювати заборонено.

Величина, яка в різні моменти часу може набувати різних значень, називається **змінною.**

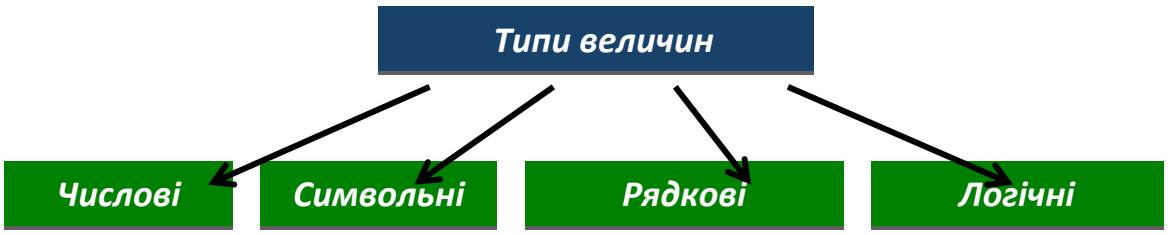
Під час виконання програми в кожний момент часу величина, як правило, має деяке значення, яке називається **поточним** значенням.

При цьому змінна величина може мати лише одне значення або не мати жодного. У процесі виконання програми величині може бути не надано ніякого конкретного значення. Тоді величина залишається невизначеною.

Тип величини — це сукупність множини допустимих значень і операцій, які дозволяється виконувати над цими значеннями.

Тип величини визначає обсяг пам'яті, необхідний для зберігання її значень, а також структуру даних.

Тип величини характеризує як постійні, так і змінні величини.



Числові величини — це величини, які можуть набувати значень з деяких числових множин. Наприклад, ціла числова величина А може набувати довільних значень із множини цілих чисел.

Приклади: 10, -2.7.

Символьні величини можуть набувати значень із деякої множини символів, і кожне значення може містити лише один символ.

Приклади: ‘a’, ‘sum’.

Рядкові величини — це величини, що можуть набувати значень із деякої множини послідовностей символів, зокрема, слів або наборів слів. Наприклад,

Приклади: ‘понеділок’, ‘вівторок’.

Симивольні та рядкові величини записуються в лапках.

Логічні величини можуть набувати тільки одного із двох значень:

True (істина) або False (хибність).

Під час виконання програм, написаних мовою Python, система сама визначає обсяг, який числові величини можуть займати в пам'яті комп'ютера, залежно від введеного їх значення.

§ 3. Об’єкти та змінні

В Python все - булеві значення, цілі числа, числа з плаваючою точкою, рядки, складні структури даних, функції - реалізовано як ***об’єкти.***

Що таке об’єкт?

Об’єкт можна уявити як скриньку, у якій міститься фрагмент даних. Об’єкт має тип (наприклад, тип цілих чисел), який визначає, що можна зробити з цими даними.

У реальному світі скринька з написом «Книги» повідомляє нам інформацію, що в ній містяться книги (фрагменти даних), які ми можемо звідти дістати або покласти нові, але виключно книги.

Точно так само і в Python - якщо об’єкт має тип цілих чисел, ви знаєте, що зможете скласти його з іншим об’єктом, який має такий самий тип цілих чисел.

Тип також визначає, чи можна змінити значення, яке зберігається в скриньці (змінюване значення), або воно є константою (незмінне значення).

Об’єкт, значення якого неможливо змінити, можна порівняти із закритою скринькою з віконцем: ви можете побачити значення, але не можете змінити його. В рамках тієї ж аналогії, об’єкт, значення якого можна змінити, схожий на відкриту скриньку: ви не тільки можете побачити значення, яке там зберігається, а й змінити його, однак не можете змінити тип об’єкта (скриньки).

Мови програмування також дозволяють визначати змінні. Ви можете визначити

їх для використання у своїй програмі.

В Python для присвоювання змінній певного значення використовується символ =. У математиці символ « =» означає «дорівнює». У багатьох мовах програмування, включаючи Python, цей символ використовується для позначення «присвоювання».

Наприклад, вираз **a=3** означає, що на об'єкт у певній області пам'яті посилається ім'я a і звертатися до них тепер слід за цим іменем.

У наступному фрагменті програми ціле число 12 присвоюється змінній з ім’ям **a**, потім на екран виводиться значення, пов’язане в поточний момент з цією змінною:

```
>>> a = 12
>>> a
12
```

В Python змінні - це просто імена. Присвоювання не копіює значення, воно прикріплює ім’я до об’єкта, який містить дані. Ім’я - це посилання на якийсь об’єкт, а не сам об’єкт. Ім’я можна розглядати як стікер-наклейку.

Необхідно дотримуватись певних вимог використання імен змінних. Імена змінних можуть містити тільки такі символи:

- літери в нижньому регістрі (від «a» до «z»)
- літери у верхньому регістрі (від «A» до «Z»)
- цифри (від 0 до 9)
- нижнє підкреслення (_)

Імена не можуть починатися з цифри.

Python особливо обробляє імена, які починаються із символа нижнього підкреслення.

Коректними є такі імена: a, a5, a_b_c_81, abc, _3a

Наступні імена є некоректними: 2 2a 2_

Не слід використовувати наступні слова, для імен змінних, оскільки вони є зарезервованими словами Python (Таблиця 2):

False	**class**	**finally**	**is**	**return**
None	**continue**	**for**	**lambda**	**try**
True	**def**	**from**	**nonlocal**	**while**
and	**del**	**global**	**not**	**with**
as	**elif**	**if**	**or**	**yield**
assert	**else**	**import**	**pass**	**break**

Таб.2

Ці слова і деякі знаки пунктуації використовуються у синтаксисі мови Python.

Тестові завдання для перевірки знань

1. Послідовність команд, записаних спеціальною мовою і призначених для виконання комп'ютером – це …
    a) Інструкція;
    b) Алгоритм;
    c) Програма;
    d) План.
2. Якого виду програмування не існує?
    a) Функціональне;
    b) Логічне;
    c) Процедурне;
    d) Об’єктивне.
3. Назвати два види величин.
    a) Початкові та кінцеві;
    b) Постійні і змінні;
    c) Введені та виведені;
    d) Визначені та невизначені.

4. Якого типу величин не існує?
    a) Рядковий;
    b) Символьний;
    c) Послідовний;
    d) Логічний.
5. Обрати перелік числових величин.
    a) True, False;
    b) 1024, 225;
    c) 'січень', 'березень';
    d) 'riz', 'dob'.
6. Обрати перелік логічних величин.
    a) True, False;
    b) 1024, 225;
    c) 'січень', 'березень';
    d) 'riz', 'dob'.
7. Обрати перелік символьних величин.
    a) True, False;
    b) 1024, 225;
    c) 'січень', 'березень';
    d) 'riz', 'dob'.
8. Обрати перелік рядкових величин.
    a) True, False;
    b) 1024, 225;
    c) 'січень', 'березень';
    d) 'riz', 'dob'.

§ 4. Введення та виведення величин

Операції та операнди

Можна сказати, що операція - це виконання певних дій над даними (операндами). Для виконання конкретних дій потрібні спеціальні інструменти —

оператори.

Наприклад, символ "+" по відношенню до чисел виконує операцію додавання, а по відношенню до рядків - конкатенацію (з'єднання).

Команди введення та виведення

Для введення та виведення величин використовують певні команди. У мові Python це команди:

- **Input-** введення з стандартного пристрою введення;
- **Print**-виведення на стандартний пристрій виведення.

Оператори (Таблиця 3)

Оператор	Назва	Пояснення	Приклади
+	Плюс	Додає два об’єкта. А також конкатенація для типу даних string	3+5 дає 8 'a'+'b' дає 'ab'
-	Мінус	Віднімає одне число від іншого або дає число зі знаком -.	50-24 дає 26 -5.2
*	Множення	Це або операція множення або операція "розмноження" для рядкового типу. Повертає рядок повторений задану к-ть разів	2*3 дорівнює 6. 'la'*3 дає 'lalala'
**	Піднесення до степені	Повертає x в степені y	3 ** 4 дає 81 (іншими словами 3

			*3 * 3 * 3)
/	Ділення	Розділити x на y	4/3 дає 1.3333333333333333
//	Цілочислове ділення	Повертає цілочисельну частку від ділення	4//3 дає 1

Таб.3

Розглянемо декілька прикладів використання цих команд.

1. Проект **Привітання**

Напишемо програму, яка виводить на екран привітання з користувачем.

Така програма буде мати вигляд:

```
print (‘Hello, School 21’)
```

2. Проект **Діалог**

Напишемо програму, яка запитує у користувача його ім’я, а потім виводить на екран привітання, називаючи користувача. Така програма буде мати вигляд:

```
print (‘Hello, School 21’)
print(‘What is your name?’)
name=input()
print (‘Hello,’name)
```

3. Проект **Сума**

Напишемо програму, за допомогою якої можна отримати результати знаходження суми чисел: 25+10.

```
input a
input b
s=a+b
print s
```

Завдання для самостійного виконання

І рівень

1. Написати програму, яка виводить на екран ваше ім’я.
2. Написати програму, яка виводить на екран ваше прізвище, ім’я та по батькові в трьох стрічках.

ІІ рівень

3. Написати програму, яка обчислює різницю двох чисел.
4. Написати програму, яка обчислює добуток двох чисел.
5. Написати програму, яка обчислює частку двох чисел.
6. Написати програму, яка обчислює суму трьох чисел.
7. Написати програму, яка обчислює значення виразу a^2.
8. Написати програму, яка обчислює периметр трикутника.
9. Написати програму, яка обчислює периметр прямокутника.
10. Написати програму, яка обчислює периметр квадрата.
11. Написати програму, яка обчислює площу прямокутника.
12. Написати програму, яка обчислює площу квадрата.

ІІІ рівень

13. Написати програму, яка виводить на екран інформацію про користувача: його ім’я, вік та місце проживання.
14. Дано діаметр кола d. Написати програму, яка обчислює довжину кола за відомим діаметром. $L=\pi \cdot d$. В якості значення π використовувати 3,14.
15. Дано довжину ребра куба a. Написати програму, яка обчислює об’єм куба $V=a^3$ і площу його поверхні $S=6 \cdot a^2$.
16. Дано довжини ребр a, b, c прямокутного паралелепіпеда. Написати програму, яка обчислює його об'єм $V=a \cdot b \cdot c$ і площу поверхні $S=2 \cdot (a \cdot b+b \cdot c+a \cdot c)$.
17. Дано два числа a і b. Написати програму, яка обчислює їх середнє арифметичне: $(a + b)/2$.
18. Знайти довжину кола L і площу круга S заданого радіуса R. $L=2 \cdot \pi \cdot R$, $S=\pi \cdot R^2$. В якості значення π використовувати 3,14.

19. Написати програму, яка обчислює довжину кола L і площу круга S заданого радіуса R. $L=2\cdot\pi\cdot R$, $S=\pi\cdot R^2$. В якості значення π використовувати 3,14.

Завдання для розумників

20. Дано два кола із загальним центром і радіусами R_1 і R_2 ($R_1 > R_2$). Знайти площі цих кіл S_1 і S_2, а також площу S_3 кільця, зовнішній радіус якого дорівнює R_1, а внутрішній радіус дорівнює R_2: $S_1=\pi\cdot(R_1)^2$, $S_2=\pi\cdot(R_2)^2$, $S_3=S_1 - S_2$. Написати програму для розв'язання даної задачі.

§ 5. Розв'язування математичних задач

Мова Python має можливість розв'язувати математичні задачі. Створимо декілька програм для розв'язання задач.

Проект 1

В одному шматку **a** *м дроту, а в іншому на* **b** *м менше.*

Скільки метрів дроту в двох шматках разом?

Математичний розв'язок:

1. Скільки метрів дроту в другому шматку? **a - b**
2. Скільки метрів дроту в двох шматках разом? **(a – b)+a**

Програма буде мати вигляд:

```
a=input ()
b= input ()
s=(a-b)+a
print s
```

Складемо таблицю перевірки для даної задачі.

a	*b*	*s*
150	**20**	**280**
380	**110**	**650**
37	**15**	**59**

Проект 2

Першого дня туристи подолали **a** *км маршруту, другого дня – вдвічі більше. Третього дня вони пройшли на* **b** *км менше, ніж другого. Скільки кілометрів подолали туристи за три дні??*

Математичний розв'язок:

1. Скільки кілометрів подолали туристи другого дня? 2**a**

2. Скільки кілометрів подолали туристи третього дня? **2a – b**

3**.** Скільки кілометрів подолали туристи всього? **a+2a+2a – b**

4**.** Спростимо вираз: **a+2a+2a – b=5a – b**

Програма буде мати вигляд:

```
a=input ()
b= input ()
L=5*a-b
print L
```

Складемо таблицю перевірки для даної задачі.

a	***b***	***L***
15	**10**	**65**
28	**15**	**125**
33	**20**	**145**

Завдання для самостійного виконання

Розв’язати задачі, написати програми для їх розв’язання та скласти таблицю виконання до них.

І рівень

1. У перший день бригада зібрала ***a*** кг картоплі, а в другий день в **2** рази більше, ніж у перший. Скільки кілограмів картоплі зібрала бригада за два дні?
2. У перший день бригада зібрала ***a*** кг картоплі, а в другий день в **2** рази більше, ніж у перший. На скільки кілограмів картоплі зібрала бригада більше у другий день?
3. У перший день продали ***a*** кг яблук, в другий — ***b*** кг, а в третій день продали ***c*** кг яблук. Скільки всього яблук продали за три дні?

ІІ рівень

4. Два велосипедиста виїхали одночасно назустріч один одному з однаковою швидкістю. Через який час вони зустрінуться, якщо відстань між ними — ***a*** км, а швидкість — ***d*** км/год?

5. Відстань ***a*** м хлопчик пробіг туди й назад за ***c*** хв. З якою швидкістю біг хлопчик?
6. Велосипедист був у дорозі ***t*** год, а мотоцикліст ***k*** год. Велосипедист проїхав ***s*** км, а мотоцикліст ***d*** км. На скільки швидкість мотоцикліста більша за швидкість велосипедиста?
7. Коли в діжку влили **n** л води, то заповнили четверту її частину. Скільки літрів води вміщує бочка?
8. ***C*** однакових автобусів за ***n*** рейсів перевезли ***k*** пасажири. Скільки пасажирів перевозив один автобус за один рейс?

III рівень

9. Маса дев'яти однакових гусей ***m*** кг. Яка маса індички, якщо відомо ,що вона на ***a*** кг більша, ніж маса гуски?
10. Автомобіль рухається зі швидкістю в ***n*** разів більшою, ніж кінь. Яка швидкість автомобіля , якщо кінь долає відстань ***s*** км за ***t*** год ?

Завдання для розумників

11. Ширина саду ***a*** м, а довжина у ***2*** рази більша. Третину площі саду займають фруктові дерева, а решту — ягідні кущі. Яку площу займають ягідні кущі?

§ 6. Алгоритми з повторенням для опрацювання величин

Під час розв'язування багатьох задач обчислювальний процес має циклічний характер, тобто частина операторів багаторазово виконується при різних значеннях змінних. Для організації повторення дій (циклів) під час запису алгоритмів використовуються оператори циклу.

Алгоритми (програми), що містять команду циклу, називаються **циклічними.** Кожна циклічна команда має тіло циклу - деяку серію команд, яку інтерпретатор повторює. Крім того, кожна циклічна команда має умову виконання повторень. Що таке умова?

Умова - це вислів, про який можна сказати виконується він чи ні. Якщо умова виконується, то говорять, що вислів істинний, правильний. Якщо ж умова не

виконується, то вислів є хибним, неправильним.

В мовах програмування умова записується за допомогою операцій порівняння. На Python вони записуються так:

1. == (дорівнює),
2. <> (не дорівнює),
3. > (більше),
4. < (менше),
5. >= (більше або дорівнює),
6. <= (менше або дорівнює).

Розглянемо, який загальний вигляд команди циклу на алгоритмічній мові та мові Python:

Алгоритмічна мова	**Мова Python**
поки умова пц тіло циклу кц	**while** умова : тіло циклу

Зверніть увагу на двокрапку після умови. Це невід'ємна частина команди циклу на мові Python.

Розглянемо приклад.

Проект 1. Знайти суму перших 5 чисел.

1. Складемо алгоритм розв’язання даної задачі.

На початку деяка величина ***s*** дорівнює нулю, оскільки не було додано жодного числа. Лічильник ***k*** теж дорівнює нулю. Ми будемо збільшувати значення лічильника до тих пір, поки воно не стане рівним 5. Після кожного збільшення значення лічильника відбувається збільшення значення суми на величину ***k.***

```
s=0
k=0
поки k<=5
    пц
```

s=s+k

k=k+1

кц

вивести s

2. Складемо таблицю перевірки умови

k	Перевірка умови	s
0	+	0
1	+	1
2	+	3
3	+	6
4	+	10
5	+	15
6	-	-

3. Напишемо програму розв’язання даної задачі мовою програмування.

```
n=input('vvedit kilkist chysel-')
s=0
k=0
while k<=n:
    s=s+k
    k=k+1
print s
```

Завдання для перевірки знань

Із наведених речень виділи ті, які є висловлюваннями. Встанови їх істинність.

1. Слоненя схоже на кенгуру.
2. Їй сподобався крокодил.
3. Вона любить математику.

4. Значення 10 не перевищує 12.
5. Вірш вивчено напам’ять.
6. Всі дельфіни не є рибами.
7. У неділю буде хороша погода.
8. Палити шкідливо.
9. Котра година?
10. Сонце обертається навколо Землі.

Завдання для самостійного виконання

Розв’язати задачі, написати програми для їх розв’язання та скласти таблицю виконання до них.

1. Знайти суму перших десяти натуральних чисел.
2. Знайти суму перших дванадцяти натуральних чисел.
3. Знайти добуток перших трьох натуральних чисел.
4. Знайти добуток перших шести натуральних чисел.
5. Знайти добуток перших десяти натуральних чисел.

§ 7. Алгоритми з розгалуженням для опрацювання величин

Сьогодні ми познайомимось із однієї із базових алгоритмічних структур – структурою розгалуження. Якщо спробувати прослідкувати за поведінкою людини протягом дня, то з'ясується, що вона майже ніколи не діє за лінійним алгоритмом. Весь час людина аналізує ситуацію, змінює свою поведінку та свої плани, пристосовується до обставин. Тому набагато частіше зустрічається такий тип алгоритму як розгалуження. Цей алгоритм обов'язково містить в собі хоча б одну умову (як правило, їх набагато більше) і виконується він в залежності від цієї умови.

При розв’язуванні задач часто використовуються алгоритми з розгалуженням, які передбачають виконання певних дій залежно від істинності деякого висловлювання, що є умовою виконання певних команд. Наприклад, алгоритм переходу дороги по пішохідному переходу, який регулюється світлофором: якщо

горить зелене світло, слід переходити дорогу, в іншому разі — слід зупинитися перед пішохідним переходом.

Алгоритмом із розгалуженням можна вважати алгоритм здійснення дзвінка з мобільного телефону: якщо є кошти на рахунку і мережа доступна, то ти набираєш номер адресата, інакше виклик не буде здійснено. При побудові таких алгоритмів використовують алгоритмічну структуру розгалуження.

Алгоритмічна структура, що дає змогу виконавцеві алгоритму вибрати сценарій подальших дій залежно від істинності певного висловлювання, називається розгалуженням.

Розрізняють дві форми структури розгалуження: повну та неповну. Структура розгалуження повної форми схожа на умовне висловлювання «Якщо — то — інакше», у якому після «то» та «інакше» записують не висловлювання, а команди, які необхідно виконати залежно від істинності висловлювання, записаного в умові. Її можна подати графічно (Мал.10):

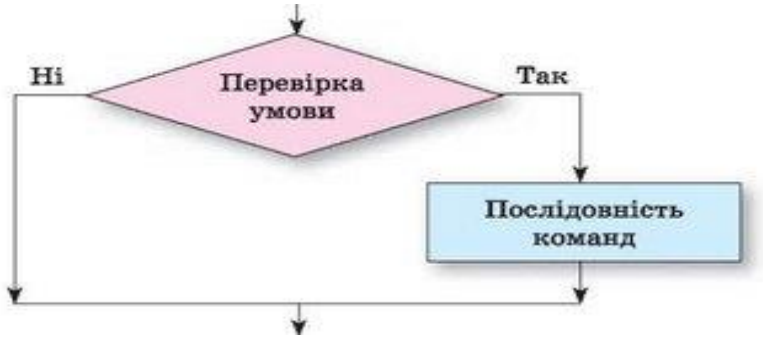
Мал.10

Наприклад, алгоритм переходу дороги можна предтавити у вигляді блок-схеми таким чином (Мал.11):

Мал.11

Скорочену форму розгалуження використовують тоді, коли деяку послідовність команд слід виконати за умови істинності висловлювання.

Структура розгалуження неповної форми схожа на умовне висловлювання «Якщо — то», у якому після «то» записують не висловлювання, а послідовність команд, які необхідно виконати, коли висловлювання, записане в умові, є істинним. Її можна подати графічно (Мал.12):



Мал.12

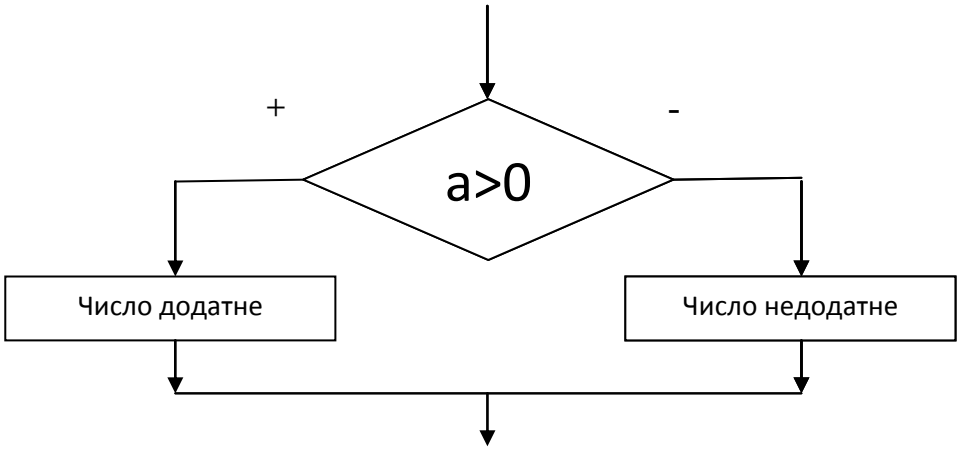
Наприклад, алгоритм планування прогулянки залежно від погодніх умов буде виглядати так (Мал.13):

Мал.13

Розглянемо приклади.

1. **Проект «Яке число?»**

Визначити тип числа, використовуючи розгалуження.



Запишемо розв’язок задачі алгоритмічною мовою.

Алг

поч

ввести **a**

якщо **a>0**

вивести «додатне»

інакше

вивести «недодатне»

кін

Мовою Python алгоритм буде представлений таким чином:

```
a=input()
if a>0:
   print"Dodatne"
else:
   print"nedodatne"
```

2. Проект «Сума додатних чисел»

Знайдемо суму додатніх чисел, якщо дано три довільних числа.

Запишемо розв’язок задачі алгоритмічною мовою.

Алг

поч

ввести **a**

ввести **b**

ввести **c**

S=0

якщо **a>0**

S=S+a

якщо **b>0**

S=S+b

якщо **c>0**

S=S+c

вивести S

кін

Мовою Python алгоритм буде представлений таким чином:

```
a=input()
b=input()
c=input()
S=0
if a>0:
   S=S+a
if b>0:
   S=S+b
if c>0:
   S=S+c
print  S
```

Завдання для самостійного виконання

Написати алгоритми та програми для розв’язання задач.

І рівень

1. Скласти програму, яка при введенні числа, меншого за 100, буде виводити на екран повідомлення «Little!», а при введенні числа, більшого за 100, буде виводити повідомлення «Large!»

ІІ рівень

2. Знайти суму від’ємних чисел із заданих трьох.
3. Знайти суму додатніх чисел, якщо дано п’ять довільних чисел.
4. Знайти добуток додатніх чисел із заданих чотирьох.

ІІІ рівень

5. Знайти суму парних чисел з ряду перших десяти натуральних чисел.
6. Знайти добуток парних чисел з ряду перших семи натуральних чисел.

7. Знайти суму непарних чисел з ряду перших п’яти натуральних чисел.
8. Знайти добуток непарних чисел з ряду перших шести натуральних чисел.

§ 8. Модуль випадкових чисел

Для генерації випадкових значень величин в мові програмування Python використовується модуль випадкових чисел **RANDOM.** Ви вже знаєте, що модуль – це самостійна програма, яка розширює можливості мови програмування.

Модуль випадкових чисел активується так само, як і графічний модуль:

from random import *

Для роботи з даним модулем познайомимося з новими командами.

1. **r = random()** - генерує число r
2. **d = randint(a, b)** - генерує координати (повертає ціле число d із проміжку [a, b]).
3. **c = choice ([k, m, n])** - обирає випадковим чином одне з перерахованих чисел: k, m чи n.

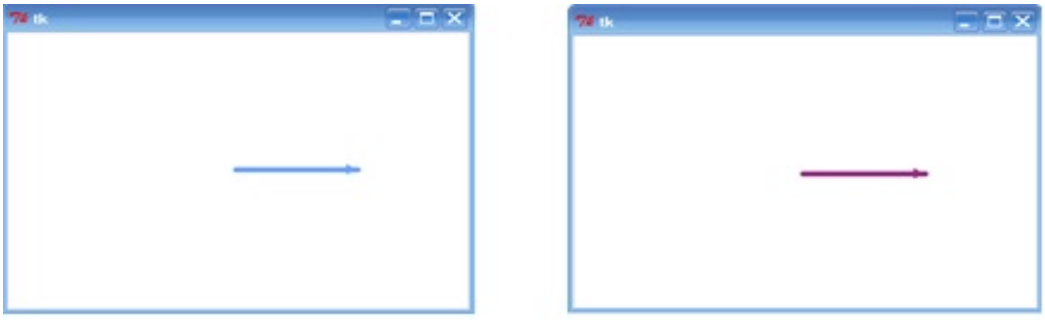
   Наприклад: c= choice ([1, 2, 3]) обирає одне з чисел: 1, 2 або 3.

Ми будемо використовувати даний модуль для створення малюнків. Тому для виконання графічних завдань потрібно підключити обидва модулі: графічний модуль та модуль випадкових чисел.

Розглянемо на прикладі, як модуль **Random** генерує кольори.

1. Проект «Лінія»

Написати програму, яка випадковим чином визначає колір лінії (Мал.14).



Мал.14

Програма	Пояснення
*from turtle import **	Підлючаємо графічний модуль
*from random import **	Підключаємо модуль випадкових чисел
red= random()	Генеруємо "кількість" червоного кольору
blue= random()	Генеруємо "кількість" синього кольору
green= random()	Генеруємо "кількість" зеленого кольору
color (red, green, blue)	Змішуємо кольори
width (5)	Задаємо товщину лінії
forward (100)	Зображуємо лінію

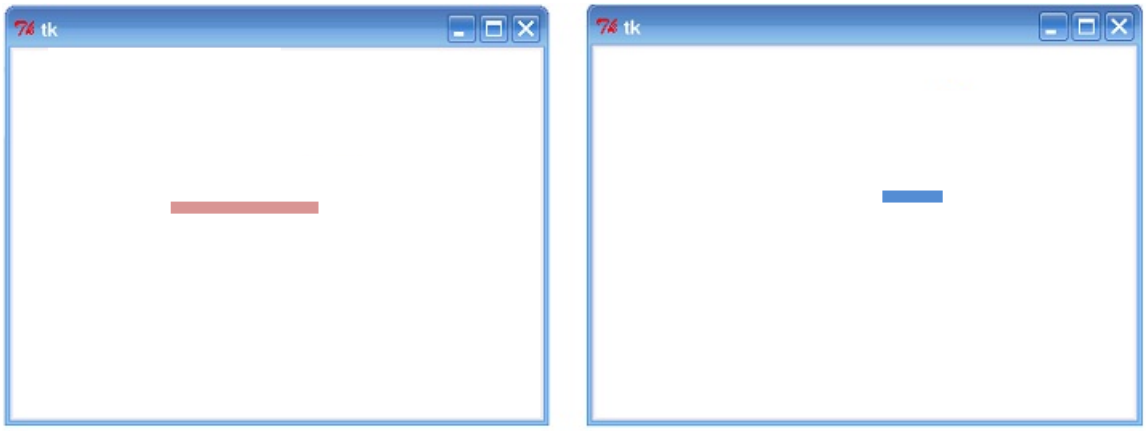
Примітка. Запустивши програму кілька раз, переконаємось, що кожного разу колір лінії інший.

Проект **№ 2**

Написати програму, що малює лінію довільної довжини.

Програма	Пояснення
*from turtle import **	Підключення графічного модуля
*from random import **	Підключення генератора випадкових чисел
width(4)	Задання товщини лінії
l= randint(-100, 100)	Генерування довжини лінії
red= random()	Генерація червоного кольору
blue= random()	Генерація синього кольору
green= random()	Генерація зеленого кольору
color (red, green, blue)	Задання кольору ліній (змішування кольорів)
forward (l)	Зображення лінії довільної довжини

Приклади виконання (Мал.15):



Мал.15

Завдання для самостійного виконання

І рівень

1. Написати програму, що малює лінію довільної товщини.
2. Написати програму, що малює прямий кут довільного кольору.
3. Написати програму, що малює коло довільного кольору.
4. Написати програму, що малює квадрат довільного кольору.
5. Написати програму, що малює коло довільного радіуса.

ІІ рівень

6. Написати програму, що малює лінію, що нахилена під довільним кутом.
7. Написати програму, що малює коло довільного кольору та довільного радіуса.
8. Написати програму, що малює лінію довільного кольору та довільної довжини.
9. Написати програму, що малює лінію довільного кольору, довільної довжини та довільної товщини.
10. Написати програму, яка малює квадрат, в якому генерується колір кожної сторони.

ІІІ рівень

11. Написати програму, що малює лінію, для якої генерується колір, товщина, довжина та кут нахилу.

12. Написати програму, що малює трикутник довільного кольору з довільною довжиною сторін.
13. Написати програму, що малює квадрат, для якого генерується колір лінії, товщина лінії, довжина сторін та кут нахилу.
14. Написати програму, що малює трикутник, для якого генерується колір лінії, товщина лінії, довжина сторін та кут нахилу.

Завдання для розумників

15. Написати програму, що малює будиночок, для якого генерується місце розташування на екрані, колір елементів та їх параметри.

§ 9. Відображення рисунків на координатній площині засобами мови програмування

Раніше ми створювали малюнки, використовуючи геометричні фігури. Але зображень, створених за допомогою графічних примітивів, досить мало. Головним недоліком таких малюнків є відсутність реалістичності. Адже як виглядатиме, наприклад, тварина, створена з самих лише ліній? Скільки потрібно буде намалювати елементів, розрахувати кутів нахилу та довжин ліній? Дуже багато.

Виконання малюнків з використанням координатної площини, зробить проект більш цікавим і привабливим. При цьому значно полегшить виконання завдання.

Для створення таких малюнків використовується команда, яка переносить «черепашку» в потрібне місце. Головне, правильно розрахувати координати потрібних точок.

goto(x,y) – перенесення черепашки в точку з координатами (x,y)

План створення програми для побудови малюнків за координатами:

1. Намалювати координатну площину, позначити осі та точки.
2. Виконати малюнок на координатній площині.
3. Визначити ключові точки.
4. Записати координати ключових точок.
5. Скласти алгоритм побудови малюнка.

6. Записати алгоритм мовою програмування.

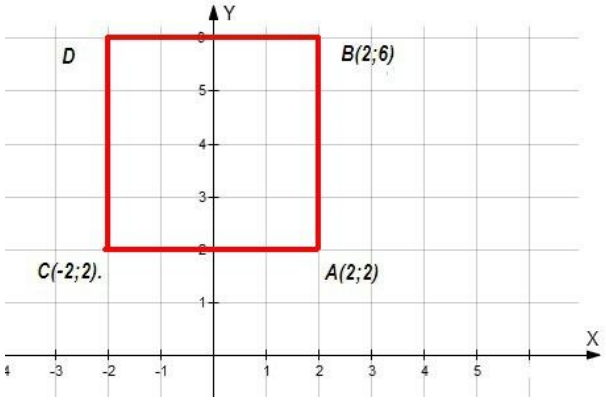
Пам'ятайте! Залити можна лише замкнену фігуру. Тому «черепашка» має пройти всі точки послідовно від першої до останньої, а потім повернутися знову в початкове положення.

Розглянемо приклади.

1. **Проект «Квадрат»**

Скласти програму, яка малює квадрат на координатній площині.

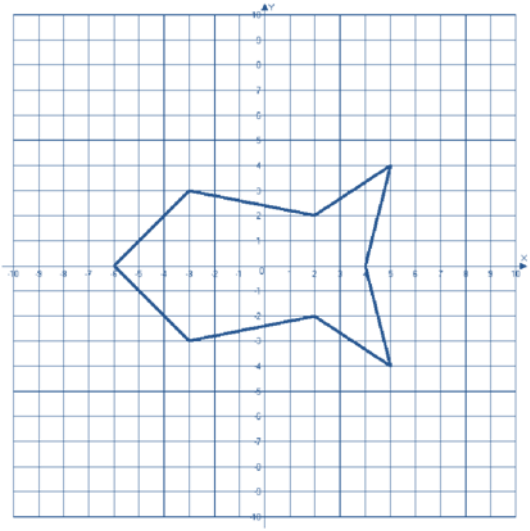
```
1. from turtle import*
2. color('red')
3. width(10)
4. up()
5. goto(-20,20)
6. down()
7. begin_fill()
8. goto(-20,60)
9. goto(20,60)
10. goto(20,20)
11. goto(-20,20)
12. end_fill()
```



2. **Проект «Риба»**

Скласти програму, яка малює рибу золотистого кольору, використовуючи задані координати.

```
1. from turtle import*
2. color('gold')
3. width(10)
4. speed(100)
5. up()
6. goto(-160,0)
7. down()
```



8. *begin_fill()*
9. *goto(-60,-60)*
10. *goto(100,-40)*
11. *goto(180,-120)*
12. *goto(160,0)*
13. *goto(180,120)*
14. *goto(100,40)*
15. *goto(-60,60)*
16. *goto(-160,0)*
17. *end_fill()*

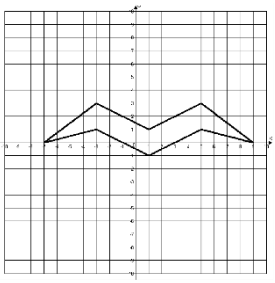
Завдання для самостійного виконання

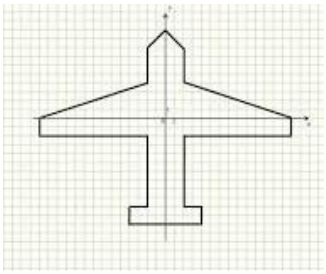
І рівень

1. Написати програму, що малює трикутник з вершинами A(-100, -50); B(-50, 100) та C(100, 20).
2. Написати програму, що малює чотирикутник з вершинами A(-200, -100); B(-100, 200); C(100,200) та D(100, -200).
3. Написати програму, що малює трикутник. Координати вершин обрати самостійно.
4. Написати програму, що малює чотирикутник. Координати вершин обрати самостійно.

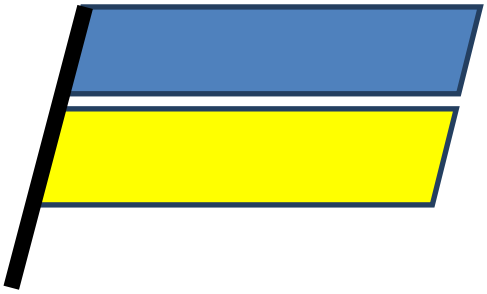
ІІ рівень

5. Написати програму, що малює трикутник з вершинами A(-100, -50); B(-50, 100) та C(100, 20) та заливає його жовтим кольором.
6. Написати програму, що малює чотирикутник з вершинами A(-200, -100); B(-100, 200); C(100,200) та D(100, -200) та заливає його фіолетовим кольором.
7. Написати програму для відображення даних малюнків.



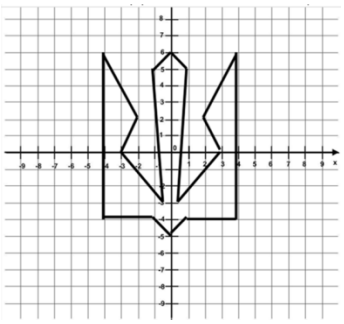


8. Написати програму,яка малює прапор України.

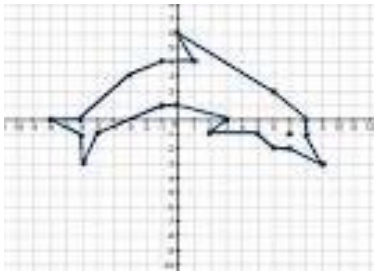


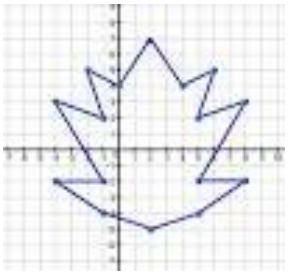
ІІІ рівень

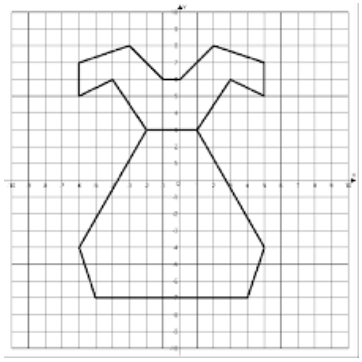
9. Написати програму, що малює герб України та заливає його золотистим кольором.

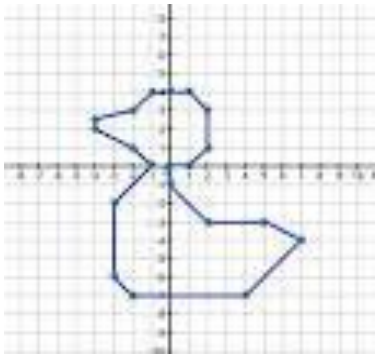


10. Написати програму, що відтворює та заливає подані малюнки.



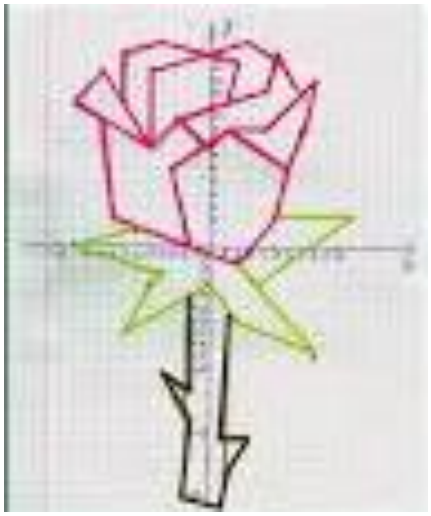


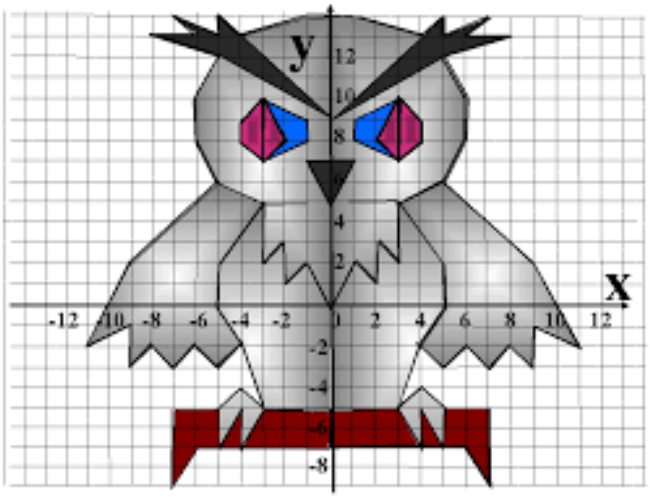




Завдання для розумників

11.Написати програму, що відтворює та заливає подані малюнки.





§ 1. Типи даних у програмуванні

Тип даних *– це набір значень та операцій, які над ними можна виконувати.*

Наприклад, числа можна множити або ділити. Але ці операції неможливо виконати над словами чи літерами.

Змінні можуть набувати значень різних типів. Розглянемо деякі стандартні типи мови Python. (Таблиця 4)

Назва типу	Позначення у Python	Приклади значень
цілий	**int**	5 1225 -4587
дробовий	**float**	3.45 -0.145 9.0
рядковий	**str (string)**	'mother' "Python" 'In 1985 year'
логічний	**bool**	Лише 2 значення: True False

Таб.4

Ви вже знаєте, що в Python для присвоювання змінній певного значення використовується символ «=». У багатьох мовах програмування, включаючи Python, цей символ використовується для позначення «присвоювання».

x=35

В Python змінні - це просто імена. Присвоювання не копіює значення, воно прикріплює ім'я до об'єкта, який містить дані. Ім'я - це посилання на якийсь об'єкт, а не сам об'єкт. В Python, якщо ви хочете дізнатися тип якогось об'єкта (змінної або значення), вам слід використовувати конструкцію **type**(об'єкт).

Спробуємо зробити це для різних значень (81, 99.99, ruby) і змінних (a, b):

1. **>>> type(a)**

   'int'>

2. **>>> type(b)**
   'int'>
3. **>>> type(81)**
   'int'>
4. **>>> type(99.99)**
   'float'>
5. **>>> type('ruby')**
   'str'>

§ 2. Опрацювання величин цілого типу

Цілі числа

Будь-яка послідовність цифр в Python вважається цілим числом.

>>> 10

Не потрібно ставити на початку числа 0, бо це викличе помилку некоректний символ.

>>> 05

File "<stdin>", line 1 05 ^ SyntaxError: invalid token

Послідовність цифр вказує на ціле число. Якщо ви поставите знак + перед цифрами, число залишиться незмінним:

>>> 132

132

>>> +132

132

Щоб вказати на від'ємне число, поставте перед цифрами знак -:

>>> -321

-321

За допомогою Python можна виконувати арифметичні дії як зі звичайним калькулятором:

>>> 25 + 9

34

>>> 145 - 37

108

>>> 8 + 3 - 2 + 1 – 106

- 96

>>> 6 * 7

42

Операцій ділення існує два види.

1. За допомогою оператора «/» виконується ділення з плаваючою точкою (десяткове ділення). Навіть, якщо ви ділите ціле число на ціле число, оператор / дасть результат з плаваючою точкою:

   >>> 9 / 5

   1.8

2. Цілочисельне ділення за допомогою оператора «//» дає цілочисельну відповідь, відкидаючи залишок:

   >>> 9 // 5

   1

   Ділення на нуль з допомогою будь-якого оператора викличе помилку:

   >>> 6 / 0

   Traceback (most recent call last): File "<stdin>", line 1, in <module> ZeroDivisionError: division by zero

   >>> 8 // 0

   Traceback (most recent call last): File "<stdin>", line 1, in <module> ZeroDivisionError: integer division or modulo by zero

   В Python вираз, який стоїть справа від знака присвоювання =, обчислюється в першу чергу, запам’ятовується результат обчислення і тільки потім результат обчислення присвоюється змінній, яка стоїть з лівої сторони. Арифметичні оператори можуть використовуватися разом із оператором присвоювання, розміщуючи їх перед символом присвоювання:

>>> a = 95 >>> a -= 3	Аналогічно виразу a = a - 3.

>>> a 92	

>>> a 92
>>> a += 8
>>> a 100
	Аналогічно виразу a = a + 8.

>>> a 100
>>> a *= 2
>>> a 200
	Аналогічно виразу a = a * 2.

>>> a 200
>>> a /= 3
>>>a 66.66666
	Аналогічно виразу a = a / 3.

За допомогою символу **%,** коли він знаходиться між двома числами, обчислюється остача від ділення першого числа на друге:

>>> 23 % 6

5

Перетворення типів: функція int()

Для того, щоб змінити інші типи даних на цілочисельний тип, слід використовувати функцію **int().**

Функція int() зберігає цілу частину числа і відкидає будь-який залишок.

Перетворення числа з плаваючою точкою в ціле число просто відсікає все, що знаходиться після десяткової точки:

>>> int(98.6)

98

>>> int(1.5e4)

15000

Розглянемо приклад перетворення текстового рядка, який містить тільки цифри або цифри і знаки + і -:

>>> int('99')

99

>>> int('-23')

-23

>>> int('+12')

12

Якщо ви спробуєте перетворити щось не подібне на число, згенерується помилка:

>>> int('22 abc')

Traceback (most recent call last): File "<stdin>", line 1, in <module> ValueError: invalid literal for int() with base 10: '22 abc'

Для встановлення пріоритетів виконання операторів можна також скористатися дужками:

>>> 7 * (3 + 4)

49

Числа з плаваючою точкою

В Python числа, що мають дробову частину, називаються дійсними (або «числами з плаваючою точкою»). Числа з плаваючою точкою обробляються так само, як і цілі: можна використовувати оператори +, -, *, /, //, **, %.

Завдання для перевірки знань

1. Співставте назву типу мовою Python та приклад значення.

1. str (string)	A. False
2. int	B. -1239
3. float	C. 5.03674
4. bool	D. “56 kg”

2. Співставте операцію та її позначення мовою Python.

1. додавання	A. **
2. віднімання	B. /
3. множення	C. *
4. ділення	D. +
5. піднесення до степеня	E. -

3. Оберіть правильні імена змінних.
    a. masa-kg
    b. days
    c. H987
    d. 7cifr
4. Оберіть правильні імена змінних.
    a. temperatyra gr
    b. zrist&&sm
    c. zrist_sm
    d. chuslo_10sm
5. Виведіть результат виконання фрагменту програми.

```
T= 20
t= 15
x= 7*T+3
print x
```

6. Виведіть результат виконання фрагменту програми.

```
c= 10
d= 12
z= 4*d+c**3
print z
```

7. Яке значення буде записане в змінну h після виконання у мові Python команди h=14.0/2+11/2?
    a. 12.5
    b. 12.0
    c. 12
8. Як потрібно записати мовою Python математичну формулу $d = 14 - \frac{7}{n+5}$, щоб отримати правильний дробовий результат?
    a. d= 14 – 7.0/(n+5)
    b. d= 14 – 7/(n+5)

c. d= 14 – float(7/(n+5))

d. d= 14 – 7/float(n+5)

9. Як записуватиметься математична формула $f = 4 + a^7 - 9a$ мовою Python?

10. У магазин привезли K ящиків із печивом по M кг у кожному. Яка загальна вага печива в кілограмах? Розставте запропоновані команди програми у правильному порядку.

1. Дія 1	A. M=input('M=')
2. Дія 2	B. print v
3. Дія 3	C. v=K*M
4. Дія 4	D. K=input('K=')

§ 3. Опрацювання величин дійсного типу

В Python числа, що мають дробову частину, називаються дійсними (або «числами з плаваючою точкою»). Числа з плаваючою точкою обробляються так само, як і цілі: можна використовувати оператори +, -, *, /, //, **, %.

Для того щоб перетворити інші типи в тип float (так називають дійсні числа в Python), слід використовувати функцію float(). Функція float() перетворює значення інших типів у значення з плаваючою точкою.

1. Булеві значення обробляються як невеликі числа:

```
>>> float(True)
1.0
>>> float(False)
0.0
```

2. Перетворення значення типу int в тип float:

```
>>> float(98)
```

98.0

3. Перетворення рядків, що містять символи, які є коректним числами (цілими або з плаваючою точкою):

>>> float('99')

99.0

>>> float ('98.6')

98.6

>>> float ('-1.5')

-1.5

>>> float ('1.0e4')

10000.0

§ 4. Опрацювання величин рядкового типу

Вважають, що програмісти добре знаються в математиці, тому що працюють з числами. Насправді, більшість програмістів працюють з **текстовими рядками** набагато частіше, ніж з числами.

Створення рядків і функція print

Будь-яка послідовність символів, укладена в лапки, в Python вважається **рядком**, при цьому рядки можуть бути укладені як в одинарні, так і в подвійні лапки:

>>> 'This is a string.'

'This is a string.'

>>> "This is also a string."

'This is also a string.'

Python 2 дозволяє брати або не брати аргументи функціїї print в дужки. Для гілки 3 дужки обов'язкові.

Навіщо мати два види лапок? Основна ідея полягає в тому, що ви можете створювати рядки, що містять лапки або апостроф. Тобто, усередині одинарних лапок можна розташувати подвійні і навпаки.

>>> 'I told my friend, "Python is my favorite language!"'

Можна також використовувати три одинарні (''') або три подвійні лапки (""""):

>>> '''Yes''' 'Yes'

>>> """No""" 'No'

Потрійні лапки не дуже корисні для таких коротких рядків. Вони, зазвичай, використовуються для того, щоб створити багаторядкові рядки, на зразок рядків вірша:

Усередині потрійних лапок символи **переходу на новий рядок** *(***\n***), пропуски зберігаються.*

```
>>> """In the age of high technology
... Without programs you can not do,
... Programmers daily
... It makes life easier for us!"""
```

Щоб змінити розділювач (за замовчуванням, це пропуск) між елементами, які виводить функція print(), треба використати її необов'язковий параметр **sep** і вказати розділювач:

```
>>>print 'Python', 'is', 'my', 'favorite', 'language!',sep=',
'Python,is,my,favorite,language!
```

Поглянемо на використання необов'язкового параметра **end** функції print. Для прикладу, запустимо наступну програму, яка збережена у окремому файлі з певним ім'ям:

```
print 'Hello' print('World')
```

Результат виведення буде таким:

```
Hello World
```

Рядки відображаються на екрані на окремих рядках виведення, оскільки функція **print()** автоматично додає символи нового рядка \n в кінець рядка, який їй передається. За необхідністю, замість символа нового рядка \n можна використовувати інший рядок, вказаний з допомогою параметра end.

Наприклад для програми:

```
print'Hello', end='' print'World'
```

виведення буде таким:

```
HelloWorld
```

В даному випадку текст виводиться в один рядок, тому що відсутній символ нового рядка \n після рядка 'Hello'. Замість символа нового рядка \n виводиться порожній рядок ' '. Цей прийом буде в нагоді в тих випадках, коли необхідно відмінити використання символа нового рядка \n, який автоматично додається в

кінці кожного виведення за допомогою функції print.

Згаданий вище, порожній рядок можна утворити за допомогою різних лапок:

>>> '' ''

>>> "" ''

>>> """""" ''

>>> '''''' ''

Необхідність створення порожніх рядків виникає, наприклад, при утворенні рядків з інших рядків:

>>> score = 55 >>> base = ''

>>> base += 'checking account: '

>>> base += str(score)

>>> base 'checking account: 55'

§ 5. Розділення та об'єднання рядків

1. Розділення рядків

Функція **split()** розбиває рядок на окремі рядки і розміщує їх у списку. Використовуючи цю функції, необхідно вказати символ-розділювач (у нашому прикладі ,):

>>> dolist = 'pass algorithm, write a program, test program'

>>> dolist.split(',')

['pass algorithm', ' write a program', ' test program']

Якщо ви не вкажете символ-розділювач, функція split() буде використовувати будь-яку послідовність пропусків, а також символи нового рядка і табуляцію:

>>> dolist.split() ['pass', 'algorithm,', 'write', 'a', 'program,', 'test', 'program']

Для виклику функції split(), навіть без аргументів, все одно потрібно додавати круглі дужки - саме так Python дізнається, що викликається функція.

2. Об'єднання рядків

Функція **join()** є протилежністю функції **split()**.

Функція **join()** об'єднує список рядків в один рядок. Виклик функції виглядає трохи заплутано, оскільки спочатку ви вказуєте рядок, який об'єднує інші, а потім - список

рядків для об’єднання: **рядок.join(список рядків)**.

Завдання для самостійного виконання

І рівень

1. Записати тип змінної b після виконання дій:

```
int a = 1;
float b = 2;
b = a
```

2. Нехай a=5, b=4. Який буде результат при виконанні таких дій?

А.
```
a=input()
b=input()
print a/b
```

Б.
```
a=input()
b=input()
print float(a/b)
```

В.
```
a=input()
b=input()
print a/b+0.0
```

Г.
```
a=float(input())
b=float(input())
print a/b
```

ІІ рівень

3. Записати програму, яка введене ціле число виводить на екран як дійсне.
4. Записати програму, яка введене дійсне число виводить на екран як ціле.
5. Поділити два цілих числа. Вивести їх результат із знаками після коми.
6. Було N діток. Бабуся спекла M пиріжків. Скільки цілих пиріжків дістанеться кожному?
7. У діда було N м дроту. Внуки-шибеники скрізь шукали пригод. У майстерні діда, де зберігалися гострі інструменти, було P дверей. Скільки дроту пішло у діда на закріплення ручки одних дверей, щоб вони міцно трималися і не відкривалися? Поекспериментуйте із цілими та дійсними числами.
8. Що виведе програма після виконання дій, якщо a=5.6. Чому?

```
a=int(input())
b=float(a)
if a==b:
   print ('yes')
else:
```

```
    print ('no')
```

9. Що виведе програма після виконання дій, якщо a=5.6. Чому?

```
a=float(input())
b=int(a)
if a==b:
    print ('yes')
else:
    print ('no')
```

10. Чому виникає помилка при спробі обчислення поданого нижче виразу? Як виправити цю помилку?

```
print('I got' + 12 + 'points')
```

III рівень

11. Перевірити, яких значень набудуть вирази «True» або «False» при перетворенні в цілі числа.
12. Перевірити, яких значень набудуть вирази «True» або «False» при перетворенні в дійсні числа.
13. Перевірити, яких значень набудуть вирази «True» або «False» при перетворенні в рядкові величини.

Завдання для розумників

14. Виконайте алгоритм в середовищі програмування:
    a. Змінній var_int надайте значення 10, var_float - значення 8.4, var_str - "No".
    b. Змініть значення, збережене у змінній var_int, збільшивши його в 3.5 рази, результат зв'яжіть зі змінною big_int.
    c. Змініть значення, збережене у змінній var_float, зменшивши його на одиницю, результат зв'яжіть з тією ж змінною.
    d. Розділіть var_int на var_float, а потім big_int на var_float. Результат даних виразів не прив'язуйте ні до яких змінних.
    e. Змініть значення змінної var_str на "NoNoYesYesYes". При формуванні нового значення використовуйте операції конкатенації (+) і повторення рядка (*).

Виведіть значення всіх змінних

§ 6. Вікна повідомлень

Графічний інтерфейс користувача (англійською GUI, Graphical user interface) — тип інтерфейсу, який дозволяє користувачам взаємодіяти з електронними пристроями через графічні зображення та візуальні вказівки, на відміну від текстових інтерфейсів, заснованих на використанні тексту, текстовому наборі вказівок і текстовій навігації. При створенні програм з графічним інтерфейсом користувача важливими є не лише алгоритми опрацювання даних, але й розробка зручного для користувача інтерфейсу програми. Потрібно передбачити взаємодію з програмою за допомогою різних кнопок, меню, піктограм, введення даних у спеціальні поля, списки для вибору значень тощо. Відповідні зображення інколи називають **віджетами** (widget — англійською штучка). Для мови програмування Python такі віджети включено у спеціальну бібліотеку tkinter. Потрібно імпортувати бібліотеку у програму, щоб використовувати її складові, створюючи графічний інтерфейс і вікна повідомлень.

Етапи створення програми з графічним інтерфейсом користувача

1. **Імпорт модуля Tkinter** можна здійснити таким чином:

   from Tkinter import *

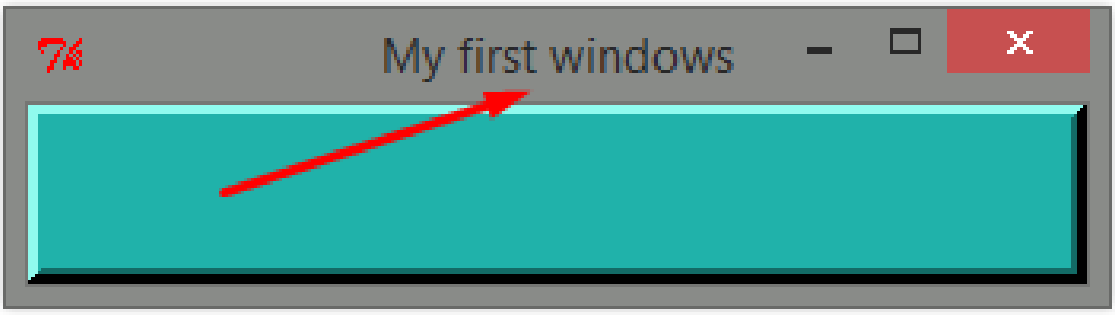
   Надалі ми будемо користуватися цим способом, бо це дає можливість не вказувати щоразу назву модуля при зверненні до об'єктів, які він містить. У версії Python 3 назву модуля пишуть з малої літери (tkinter), хоч у попередніх версіях використовують велику (Tkinter).

2. **Створення головного вікна**. В сучасних операційних системах будь-яку клієнтську програму подано вікном. Його можна назвати головним, бо в ньому розташовують всі інші віджети. Об'єкт вікна верхнього рівня створюють при зверненні до класу Tk модуля Tkinter. Змінну, пов'язану з об'єктом-вікном, зазвичай називають **root** (корінь), хоча обмежень на назву немає. Другий рядок програми має такий вигляд:

   root = Tk()

Далі створюємо заголовок вікна (Мал.16). Третій рядок програми має такий вигляд:

root.title ('...')

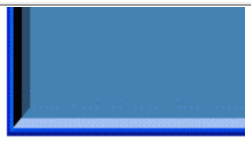
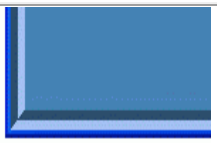
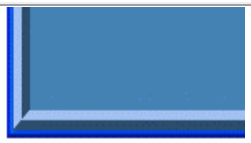


Мал.16

Побудова вікна

Рядки програми	**Пояснення**
*from Tkinter import **	підключення бібліотеки Tkinter
root = Tk()	створення вікна
root.title ('...')	назва вікна
root.config (bg = "колір_фону", width = ширина_вікна, height = висота_вікна, relief = тип_рамки, bd = товщина_рамки)	Задання конфігурації (параметрів вікна)
root.mainloop()	Запуск вікна

Типи рамок:

FLAT (плоска рамка)	**SUNKEN** (увігнута рамка)	**RAISED** (опукла рамка)	**GROOVE** (у вигляді жолобу)	**RIDGE** (у вигляді хребта)
				

§ 7. Додавання тексту до вікна повідомлень

У вікні можна розміщувати різні об'єкти. Наприклад, для розміщення тексту у

вікні використаємо клас **Label** (з англ. ярлик).

Створимо конкретний об'єкт цього класу з іменем **label.**

label = Label(root, text='Hello!')

У дужках необхідно вказати, у якому вікні буде розташований текст і який саме текст. Тому першою опцією є ім'я створеного вікна root, другий параметр text приймає значення 'Hello!'.

До об'єкту label треба застосувати метод **pack().**

label.pack()

Якщо цей метод не використати, текст у вікні не буде відображеним (не з'явиться).

Щоб додати текст до вікна, потрібно виконати певну послідовність дій.

1. Підключити бібліотеку Tkinter.
2. Створити вікно.
3. Назвати його.
4. Задати конфігурацію (параметри).
5. Створити надпис (текст) і задати його параметри.
6. Розмістити текст на вікні.
7. Запустити вікно.

Параметри:

1. ***width*** – ширина;
2. ***height*** – висота;
3. ***background(bg)*** - колір фону;
4. ***foreground(fg)*** – колір символів;
5. ***font = ('Arial', 40)*** – шрифт та розмір тексту;

Розглянемо приклади.

Задача № 1

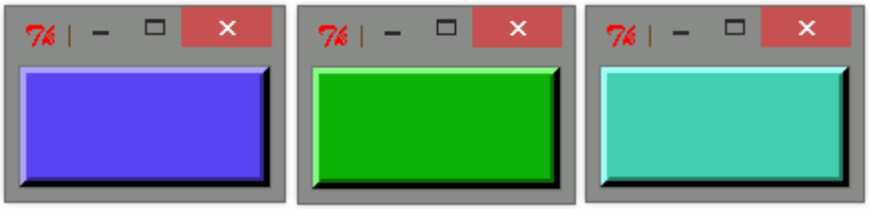
Написати програму, яка зображає вікно синього кольору з назвою “My first window”, розмірами 200x50, товщиною рамки 10 пікселів. Додати надпис “Hello!!!” жовтого

кольору, шрифт – Arial, розмір - 40.

```
from Tkinter import *
win=Tk ()
win.title ('My first windows')
win.config (width=200, height=50,bg= “blue”, relief=RAISED, bd=10)
lab1=Label (win, text='Hello!!!', fg='yellow', font = ('Arial', 40))
lab1.pack()
win.mainloop()
```

Задача № 2

Написати програму зображення вікна розмірами 40x60, назвою“My windows” з товщиною рамки 5, колір якого генерується програмою.



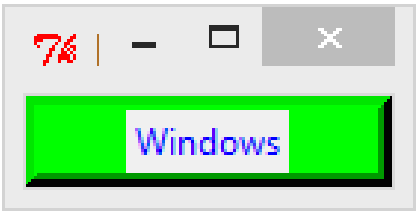
Для генерування кольору слід задати “кількість” червоного, зеленого та синього кольорів (числом у межах 1 байта). Потім змішати кольори.

```
red = randint (0, 255)
green = randint (0, 255)
blue = randint (0, 255)
col = "#%02x%02x%02x"  %(red, green, blue)
```

```
from Tkinter import *
from random import *
win=Tk ()
win.title ('My windows')
red = randint (0, 255)
green = randint (0, 255)
blue = randint (0, 255)
```

```
col = "#%02x%02x%02x"  %(red, green, blue)
win.config (width = 40, height = 60, bg = col, relief = RAISED, bd = 5)
win.mainloop()
```

Задача № 3



Написати програму, яка зображуватиме вікно із однією з 3-ох описаних надписів “Microsoft Word”, “Windows” або “Python” (вибір надпису здійснює програма). Параметри кожного тексту задати на вибір.

```
from Tkinter import *
from random import *
win=Tk()
win.title ('My windows')
win.config (width = 40, height=60, bg='green', relief=RAISED, bd=5)
lab1 = Label (win, text='Microsoft Word', fg='red', font = ('Arial',40))
lab2 = Label (win, text='Windows', fg='blue')
lab3 = Label (win, text='Python', fg='yellow', bg='green', font='Centaur')
lab = choice ([lab1, lab2, lab3])
lab.pack()
win.mainloop()
```

Завдання для самостійного виконання

І рівень

1. Встановити відповідність між командою та її значенням..

А. root = Tk()	1. Конфігурація вікна
Б. root.title (‘...’)	2. Запуск вікна
В. root.mainloop()	3. Заголовок вікна
Г. root.config	4. Створення вікна

2. Встановити відповідність між параметром та його значенням..

А. foreground(fg)	1. Колір фону
Б. font = ('Arial', 40)	2. Колір символів
В. background(bg)	3. Розмір тексту
Г. height	4. Шрифт тексту

ІІ рівень

3. Записати програму, яка створює та запускає вікно довільного кольору та розміру з назвою School.
4. Записати програму, яка створює та запускає вікно синього кольору шириною 100 пікселів та висотою 150 пікселів з назвою School.
5. Записати програму, яка створює та запускає вікно червоного кольору шириною 200 пікселів та висотою 50 пікселів з назвою Class.

ІІІ рівень

6. Написати програму зображення вікна розмірами 100x100, з назвою “My class” з товщиною рамки 10, колір якого генерується програмою.
7. Написати програму зображення вікна розмірами 200x50, з назвою “My class” з товщиною рамки 10 та написом у вікні «7-А».
8. Написати програму зображення вікна розмірами 200x200, з назвою “My window” з товщиною рамки 10 та трьома написами у вікні «Прізвище», «Ім’я», «По батькові» .

Завдання для розумників

9. Написати програму зображення трьох вікон розмірами 100x50, з назвами “My first window” , “My second window” , “My third window” з товщиною рамки 5 та трьома написами у вікнах «Прізвище», «Ім’я», «По батькові».

§ 8. Створення кнопок у вікні. Клас Button

Створимо вікно, в якому розмістимо кнопку.Для створення кнопок використовується клас **Button.** Команда, яка відповідає за створення кнопки, має вигляд:

```
b1 = Button(root, text='My button!')
```

```
b1.pack()
```

Напишемо програму, яка створює кнопку у вікні.

```
from Tkinter import *
root = Tk()
b1 = Button(root, text='My button!')
b1.pack()
root.mainloop()
```

Якщо розміри кнопки не задані, то вони визначаються текстом на ній.

Створимо вікно з двома кнопками, що будуть розташовуватися одна під одною.

```
from Tkinter import *
root = Tk()
b1 = Button(root, text='My first button!')
b1.pack()
b2 = Button(root, text='Second!')
b2.pack()
root.mainloop()
```

Ускладнимо завдання: **Нехай при натисненні на першу кнопку видається повідомлення "Hello!"**.

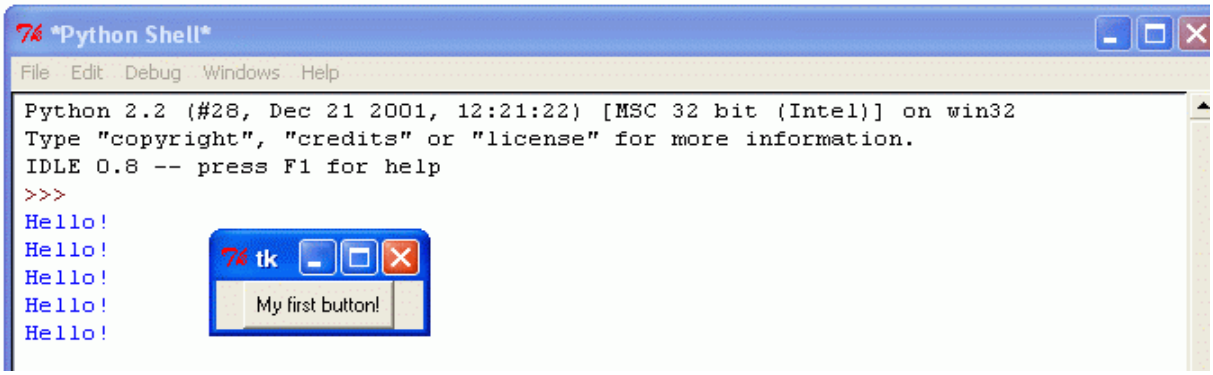
По-перше, подія пов'язана з кнопкою, тому кнопка має містити атрибут, що обробляє цю подію. Цей атрибут носить назву **command**. Яке ж може бути його значення? Яка нам знайома структура виконується лише тоді, коли ми її викликаємо? Це функція. Напишемо функцію, яка виводить дане повідомлення. Потім значенням атрибуту **command** поставимо ім'я функції. Вона буде викликатися лише при натисненні на кнопку.

```
from Tkinter import *
def hello():
   print 'Hello!'
root = Tk()
```

```
b1 = Button(root, text='My first button!', command=hello)
b1.pack()
root.mainloop()
```

Текст '**Hello!**' виводиться не у нашому вікні, а у вікні Python Shell.

Результат запуску програми і п'яти натискувань кнопки (Мал.17):



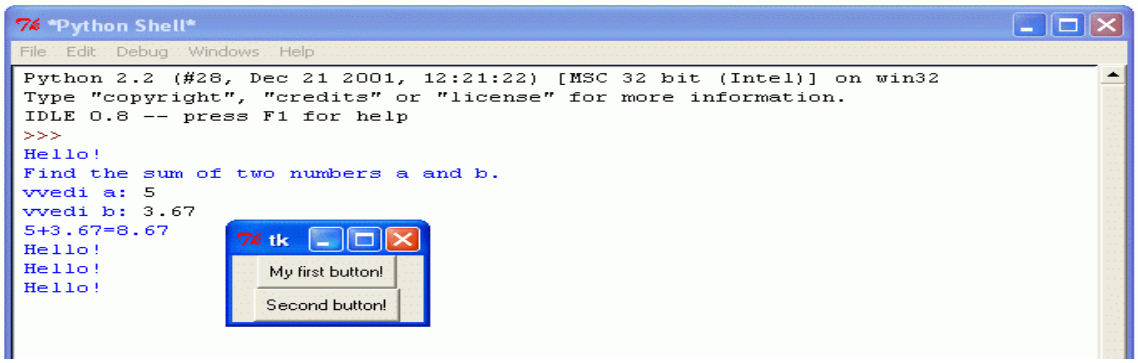
Мал.17

Розглянемо ще один приклад, у якому при натисненні на першу кнопку виводиться "Hello!", друга кнопка дозволяє обрахувати суму двох чисел.

```
from Tkinter import *
def hello():
    print 'Hello!'
def sum():
    a = input('Введи a: ')
    b = input('Введи b: ')
    S=a+b
    print S
root = Tk()
b1 = Button(root, text='My first button!', command=hello)
b1.pack()
b2 = Button(root, text='Second button!', command=sum)
b2.pack()
root.mainloop()
```

Результат запуску програми і натискування спочатку на першу, потім на другу кнопку (при цьому вводяться два числа і виводиться результат), і три рази на першу

кнопку (Мал.18):



Мал.18

Використаємо бібліотеку **turtle** для опрацювання натиснення на третю кнопку. Нехай при цьому в окремому вікні малюється коло.

При підключенні модуль turtle повинен обов'язково йти перед модулем Tkinter, бо вікно turtle написане за допомогою Tkinter.

```
from turtle import *
from Tkinter import *
def hello():
    print 'Hello!'
def sum():
    a = input('Введи a: ')
    b = input('Введи b: ')
    S=a+b
    print S
def circle():
    width(10)
    color(‘red’)
    circle(100)
root = Tk()
b1 = Button(root, text="First button!", command=hello)
b1.pack()
b2 = Button(root, text="Second button", command=sum)
b2.pack()
```

```
b3 = Button(root, text="Third button", command= circle)
b3.pack()
root.mainloop()
```

Завдання для самостійного виконання

І рівень

1. Яка команда відповідає за введення функції?
   a. Button
   b. def
   c. pack
   d. command
2. Записати команду, яка дозволяє створити кнопку з назвою «Knopka».
3. Скільки кнопок можна створити у вікні?

ІІ рівень

4. Записати програму, яка створює кнопку «Knopka» у вікні.
5. Записати програму, яка створює у вікні кнопку жовтого кольору.
6. Записати програму, яка створює у вікні дві кнопки різних кольорів.

ІІІ рівень

7. Записати програму, яка створює у вікні три кнопки різних кольорів з назвами, що позначають ваше прізвище, ім’я та по батькові.
8. Записати програму, яка створює у вікні дві кнопки різних кольорів, одна з яких рахує суму чисел, а інша – різницю.
9. Записати програму, яка створює у вікні дві кнопки різних кольорів, одна з яких малює коло довільного радіуса, а інша – трикутник.

Завдання для розумників

10. Створіть вікно з 4-ма кнопками однакового розміру. При натисненні на першу у вікні turtle черепашка малює червоний квадрат, при натисненні на другу - черепашка малює зелений трикутник, третя кнопка дозволяє черепашці намалювати жовтий круг, четверта - пурпуровий шестикутник.

Посібник з програмування для 8 класу

§ 1. Поняття класу та об’єкту

Пригадаємо два ключових поняття: Клас і Об'єкт.

Клас - це абстрактний тип даних. За допомогою класу описується деяка сутність (її характеристики і можливі дії). Наприклад, клас може описувати студента, автомобіль і т.д. Описавши клас, ми можемо створити його примірник - Об'єкт. **Об'єкт** - це вже конкретний представник класу.

Приклад

Припустимо, нам в програмі необхідно працювати з країнами. Країна - це абстрактне поняття. У неї є такі характеристики, як назва, населення, площа, прапор та інше. Для опису такої країни буде використовуватися клас з відповідними полями даних. Такі країни, як Болгарія і Україна будуть вже об'єктами (конкретними представниками типу країна).

Оголошення класу задає *представлення* об'єктів *цього класу* і *набір операцій,* які можна застосовувати до таких об'єктів.

Клас можна порівнювати з формою для випічки печива — форма одна, а печива можна випекти безліч. Печиво — це конкретні об'єкти, екземпляри класу печиво, яке може бути з різною начинкою. Поля дозволяють вмістити дані про певний реальний об'єкт, а методи здійснювати обробку цих даних.

Приклад

Наприклад, можна створити загальний клас *Людина* з полями *Ім'я* та *Прізвище*, *рік народження*, *професія*, *зарплата*. При створенні ж на основі класу конкретного екземпляру, дані поля заповнюються конкретними даними про певну людину. Обробкою цих даних можуть займатися відповідні методи. Наприклад, можна створити метод для обчислення віку людини, тощо.

§ 2. Пригадаймо підключення віконного інтерфейсу

Що таке Tkinter?

Tkinter (від англ. tk interface) - це графічна бібліотека, яка дозволяє створювати програми з віконним інтерфейсом.

```
from Tkinter import *
```

В Tkinter візуальні форми називаються віджетами (widget, від англ. window gadget) – стандартизований компонент графічного інтерфейсу, з яким взаємодіє користувач.

Клас Tk

Tk є базовим класом будь-якого Tkinter-додатку . При створенні об'єкта цього класу запускається інтерпретатор та створюється базове вікно додатку.

Мінімальний додаток на Tkinter буде таким:

```
from tkinter import *
root = Tk()
root.mainloop()
```

Всі віджети в Tkinter мають деякі спільні властивості. Розглянемо деякі з них перед вивченням конкретних випадків. Віджети створюються викликом конструктора відповідного класу.

Віджет = клас(**master** , **options** = ….)

Де:

- **master** - батьківське вікно.
- **options** - список найбільш часто використовуваних властивостей для цього віджета. Це може бути шрифт (font=...), колір віджета (bg=...), команда, що виконується при активації віджета (command=...) тощо. Приклад коду:

```
from Tkinter import *
def button_clicked():
    print (u"Клик!")
root=Tk()
# кнопка за замовчанням
button1 = Button()
```

```
button1.pack()
# кнопка з вказанням батьківського віджету та кількома аргументами
button2 = Button(root, bg="red", text=u"Кликни меня!", command=button_clicked)
button2.pack()
root.mainloop()
```

В таблиці вказано основні властивості віджетів та їх значення (Таблиця 5)

	Варіант і опис
1	**bd** Ширина межі в пікселях. За замовчуванням - 2.
2	**bg** Колір фону.
3	**Confine** Якщо значення true (значення за замовчуванням), полотно не можна прокручувати за межі scrollregion.
4	**Cursor** Курсор, який використовується на полотні, наприклад *стрілка, коло, точка тощо*
5	**Height** Розмір полотна в Y-вимірі.
6	**highlightcolor** Колір, зображений у фокусі.
7	**Relief** Рельєф визначає тип межі. Деякі значення є SUNKEN, RAISED, GROOVE і RIDGE.
8	**scrollregion** Кортеж (w, n, e, s), який визначає, наскільки велика область може бути прокручена, де w - ліва сторона, n - верхня, e - права сторона, s - нижня.
9	**Width**

	Розмір полотна в розмірності X.
10	**xscrollincrement** Якщо ви встановите цей параметр на деякий позитивний розмір, полотно можна розташувати тільки на кратній цій відстані, а значення буде використано для прокрутки одиницями прокрутки, наприклад, коли користувач натискає стрілки на кінцях смуги прокрутки.
11	**xscrollcommand** Якщо полотно прокручується, цей атрибут повинен бути методом .set () горизонтальної смуги прокрутки.
12	**ycrollincrement** Працює як xscrollincrement, але керує вертикальним рухом.
13	**yscrollcommand** Якщо полотно прокручується, цей атрибут повинен бути .set () методом вертикальної смуги прокручування.

Таб.5

§ 3. Полотно. Клас Canvas

Полотно (Canvas) - це прямокутна область, призначена для малювання картинок або інших складних макетів. На полотні можна розміщувати графіку, текст, віджети або кадри. Ось простий синтаксис для створення цього віджета

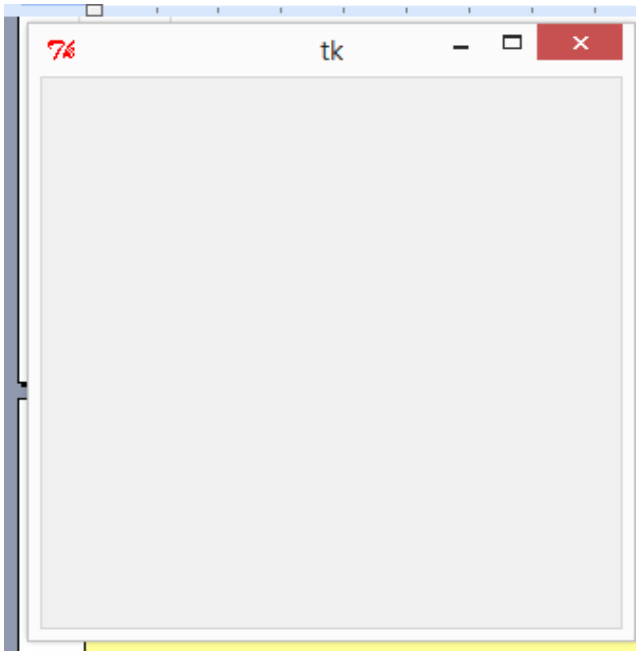
```
w = Canvas ( master, option=value, ... )
```

Для початку створимо вікно, в якому буде розміщено полотно 300 на 300 пікселів:

```
from Tkinter import * #Підключення модулю Tkinter
root = Tk() # ініціалізація графічного інтерфейсу
canvas = Canvas(root, width=300, height=300) # ініціалізація полотна розмірами 300x300
canvas.pack() # розміщення полотна у вікні модуля
root.mainloop() # створення постійного циклу
```

В результаті виконання записаних кодів матимемо сіре віконце, в якому можна

розміщувати інші елементи (Мал.19).



Мал.19

Віджет Canvas може підтримувати такі стандартні елементи -

arc - створює елемент дуги.

```
coord = 10, 50, 240, 210
arc = canvas.create_arc(coord, start=0, extent=150, fill="blue")
```

line - створює позицію лінії.

```
line = canvas.create_line(x0, y0, x1, y1, ..., xn, yn, options)
```

oval - створює коло або еліпс за заданими координатами. Він приймає дві пари координат; верхній лівий і нижній праві кути обмежувального прямокутника для овалу.

```
oval = canvas.create_oval(x0, y0, x1, y1, options)
```

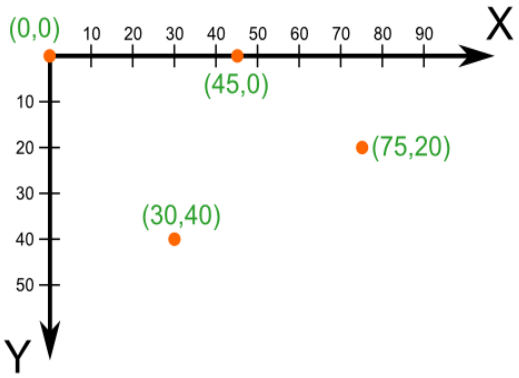
polygon - створює елемент багатокутника, який повинен містити принаймні три вершини.

```
polygon = canvas.create_polygon(x0, y0, x1, y1,...xn, yn, options)
```

rectangle - створює прямокутник. Він приймає дві пари координат: верхній лівий і нижній правий кути.

```
rectangle = canvas.create_ rectangle (x0, y0, x1, y1, options)
```

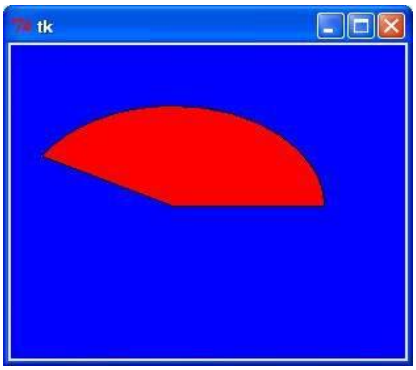
Для розміщення об’єктів на полотні слід вказати їх координати, пам’ятаючи, що точкою відліку є верхній лівий кут.



Спробуйте наступний приклад -

```
import Tkinter
top = Tkinter.Tk()
C = Tkinter.Canvas(top, bg="blue", height=250, width=300)
coord = 10, 50, 240, 210
arc = C.create_arc(coord, start=0, extent=150, fill="red")
C.pack()
top.mainloop()
```

Коли виконується вищевказаний код, він дає наступний результат (мал.20)-



Мал.20

Видалення елементів з полотна

Під час написання кодів, які створюють різні об’єкти, ми працювали таким чином:

- резервували змінну

- присвоювали їй значення об’єкту на полотні.

Тобто кожна створена фігура була прив’язана до певного ідентифікатору. Це потрібно для того, щоб надалі можна було легко ними керувати.

Наприклад, для видалення фігури з полотна використовують метод canvas.delete:

canvas.delete(square) – видалення квадрату з полотна
canvas.delete(text) – видалення тексту з полотна,

Таким чином, додаючи або видаляючи елементи з полотна можна створити інтерактивне вікно для взаємодії полотна з користувачем.

Завдання для перевірки знань

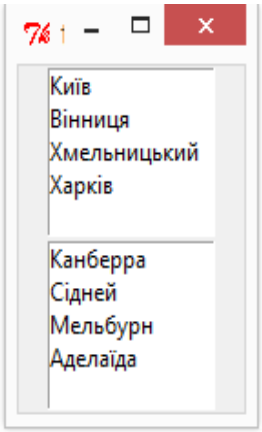
Визначити, що означають стрічки вказаного нижче коду

```
from Tkinter import *
_____________________________________________________________________
root = Tk()
_____________________________________________________________________
canvas = Canvas(root, width=300, height=300)
_____________________________________________________________________
canvas.pack(fill=BOTH)
_____________________________________________________________________
circle = canvas.create_oval(10,10,290,290, fill="#00f")
_____________________________________________________________________
diamond = canvas.create_polygon(150,10,10,150,150,290,290,150, fill="red")
_____________________________________________________________________
square = canvas.create_rectangle(80,80,220,220, fill="green")
_____________________________________________________________________
text = canvas.create_text(150,150, text="Tkinter canvas", fill="purple",
font=("Helvectica", "16"))
_____________________________________________________________________
root.mainloop()
```

§ 4. Екранні форми: список. Клас Listbox

Listbox – це віджет, який має вигляд списку, з елементів якого користувач може вибирати один чи кілька пунктів. Наприклад:

```
from Tkinter import *
root=Tk()
listbox1=Listbox(root,height=5,width=15,selectmode=EXTENDED)
listbox2=Listbox(root,height=5,width=15,selectmode=SINGLE)
list1=[u"Київ",u"Вінниця",u"Хмельницький",u"Харків"]
list2=[u"Канберра",u"Сідней",u"Мельбурн",u"Аделаїда"]
for i in list1:
    listbox1.insert(END,i)
for i in list2:
    listbox2.insert(END,i)
listbox1.pack()
listbox2.pack()
root.mainloop()
```



В Python 2.x для того, щоб відображалися українські букви, слід перед стрічкою поставити букву u.

В Tkinter спочатку створюється об’єкт Listbox, потім він заповнюється з допомогою методу insert().

```
for i in list1:
    listbox1.insert(END,i)
```

Перший аргумент в insert() це індекс місця, куди буде вставлено елемент. Якщо він має бути в кінці списку, то індекс позначають константою END.

Другий аргумент – це сам об’єкт, що вставляється.

Додаткова властивість selectmode при значенні SINGLE дозволяє обрати тільки один елемент списку, а при значенні EXTENDED – будь-яку кількість, затиснувши Ctrl або Shift.

Якщо для Listbox пострібен скроллер, то він налаштовується як і для текстового поля. В програму додається віджет Scrollbar та зв'язується з екземпляром Listbox.

З допомогою методу get() із списку можна отримати один елемент по його індексу, або зріз, якщо вказати два індекси. Метод delete() видаляє один елемент чи зріз.

Метод curselection() дозволяє отримати індекси обраних елементів з Listbox.

§ 5. Екранні форми: рамки. Клас Frame

Метод pack()

Від зручності інтерфейсу залежить зручність використання програми. Тому дуже важливо навчитися ефективно і зручно розміщувати об'єкти у вікні.

Організовуючи віджети в просторі, програміст стає і дизайнером, розробником інтерфейсів.

В Tkinter існує три способи організації простору: пакувальник, сітка та розміщення за координатами.

Пакувальник (packer) викликається методом pack(), що є у всіх віджетів-об'єктів. Без нього об'єкт не відобразиться у вікні.

Якщо в пакувальник не передавати аргументи, то віджети будуть розміщені вертикально, один над одним.

У метода pack() є параметр side (сторона), який набуває одного з чотирьох значень-констант tkinter – TOP, BOTTOM, LEFT, RIGHT (верх, низ, ліво, право). За замовчуванням, коли в pack() не вказують side, він набуває значення TOP. Через це віджети розміщуються вертикально.

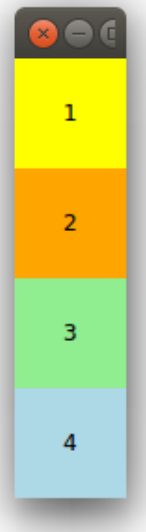
Приклад (Мал. 21)

Створимо 4 кольорові мітки та розглянемо різні комбінації значень side:

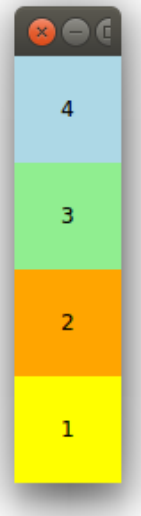
```
...
l1 = Label(width=7, height=4, bg='yellow', text="1")
```

```
l2 = Label(width=7, height=4, bg='orange', text="2")
l3 = Label(width=7, height=4, bg='lightgreen', text="3")
l4 = Label(width=7, height=4, bg='lightblue', text="4")
...
```

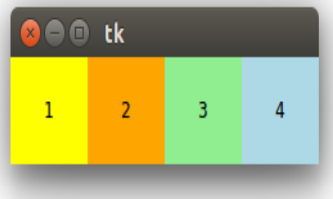
```
l1.pack()
l2.pack()
l3.pack()
l4.pack()
```



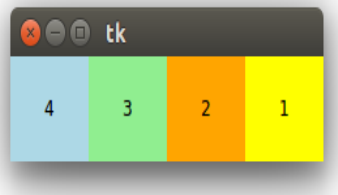

```
l1.pack(side=BOTTOM)
l2.pack(side=BOTTOM)
l3.pack(side=BOTTOM)
l4.pack(side=BOTTOM)
```



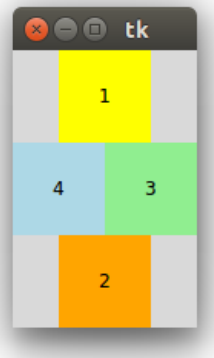

```
l1.pack(side=LEFT)
l2.pack(side=LEFT)
l3.pack(side=LEFT)
l4.pack(side=LEFT)
```




```
l1.pack(side=RIGHT)
l2.pack(side=RIGHT)
l3.pack(side=RIGHT)
l4.pack(side=RIGHT)
```




```
l1.pack(side=TOP)
l2.pack(side=BOTTOM)
l3.pack(side=RIGHT)
l4.pack(side=LEFT)
```




```
l1.pack(side=LEFT)
l2.pack(side=LEFT)
l3.pack(side=BOTTOM)
l4.pack(side=LEFT)
```

Мал.21

Якщо ж ці мітки потрібно буде розмістити квадратом: дві зверху, дві знизу, то тут стикаємося з проблемою. Для її вирішення можна скористатися віджетом класу Frame.

Віджет **Frame** (рамка) призначений для організації віджетів всередині вікна. Фрейми розміщують в головному вікні, а в фреймах – інші віджети:

```
from Tkinter import *
root = Tk()
f_top = Frame(root)
f_bot = Frame(root)
l1 = Label(f_top, width=7, height=4, bg='yellow', text="1")
l2 = Label(f_top, width=7, height=4, bg='orange', text="2")
l3 = Label(f_bot, width=7, height=4, bg='lightgreen', text="3")
l4 = Label(f_bot, width=7, height=4, bg='lightblue', text="4")
f_top.pack()
f_bot.pack()
l1.pack(side=LEFT)
l2.pack(side=LEFT)
l3.pack(side=LEFT)
l4.pack(side=LEFT)
root.mainloop()
```

Результат (Мал.22):

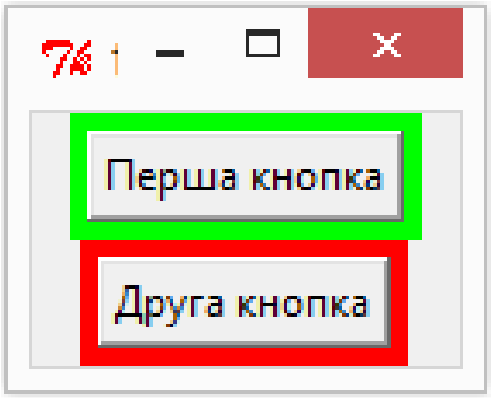
Мал.22

Ще один приклад з розміщенням кнопок у фреймах:

```
from Tkinter import *
root=Tk()
frame1=Frame(root,bg='green',bd=5)
```

```
frame2=Frame(root,bg='red',bd=5)
button1=Button(frame1,text=u'Перша кнопка')
button2=Button(frame2,text=u'Друга кнопка')
frame1.pack()
frame2.pack()
button1.pack()
button2.pack()
root.mainloop()
```

Атрибут bd відповідає за товщину межі рамки (Мал. 23).



Мал.23

§ 6. Екранні форми: перемикачі. Клас Radiobutton

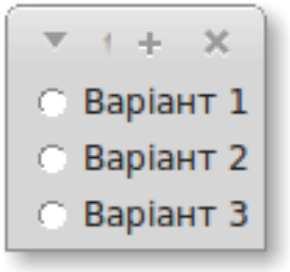
Radiobutton — *перемикач* — елемент списку з місцем для мітки за умови можливості виставити лише одну мітку. Перемикач завжди використовують у групі, причому увімкненим може бути лише один перемикач.

Реалізація цього віджета відрізняється від реалізації для Checkbutton:

```
from Tkinter import *
root=Tk()
v=IntVar()
rbutton1 = Radiobutton (root, text='Варіант 1', variable=v, value=1)
rbutton2 = Radiobutton (root, text='Варіант 2', variable=v, value=2)
rbutton3 = Radiobutton (root, text='Варіант 3', variable=v, value=3)
rbutton1.pack ()
```

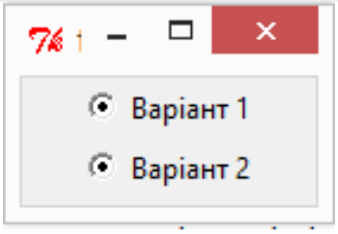
```
rbutton2.pack ()
rbutton3.pack ()
root.mainloop ()
```

У цьому прикладі використано лише одну змінну. Залежно від того, який пункт обрано, вона змінює своє значення.



Якщо буде створено два перемикача без відповідних атрибутів, то вони обидва будуть увімкнені і змінити щось буде неможливо:

```
from Tkinter import *
root=Tk()
r1 = Radiobutton (text='Варіант 1')
r2 = Radiobutton (text='Варіант 2')
r1.pack ()
r2.pack ()
root.mainloop ()
```



Ці перемикачі ніяк не зв’язані між собою та за замовчуванням – обидва увімкнені. Зв’язок встановлюється через змінну, різні значення якої відповідають увімкненню різних перемикачів групи.

У всіх кнопок однієї групи атрибут variable описує прикріплення до віджета змінної тобто набуває значення спільної для групи змінної. Атрибуту value присвоюються різні значення цієї змінної.

В Tkinter використовують лише спеціальні класи-змінні:

- ❖ BooleanVar – для булевих значень 0 або 1 та True або False,
- ❖ IntVar – для цілих значень,
- ❖ DoubleVar – для дійсних (дробових) значень,
- ❖ StringVar – для рядкових (стрічкових) значень.

Зазвичай в програмному коді необхідно зняти дані про те, який перемикач увімкнено. Для цього використовують метод get() екземплярів змінних Tkinter.

```
from Tkinter import *
 def change():
    if var.get() == 0:
        label['bg'] = 'red'
    elif var.get() == 1:
        label['bg'] = 'green'
    elif var.get() == 2:
        label['bg'] = 'blue'
 root = Tk()
 var = IntVar()
var.set(0)
red = Radiobutton(text="Red", variable=var, value=0)
green = Radiobutton(text="Green", variable=var, value=1)
blue = Radiobutton(text="Blue", variable=var, value=2)
button = Button(text="Змінити", command=change)
label = Label(width=20, height=10)
red.pack()
green.pack()
blue.pack()
```

```
button.pack()
label.pack()
 root.mainloop()
```

Перевірте роботу наведеної програми.

§ 7. Екранні форми: прапорці. Клас Checkbutton

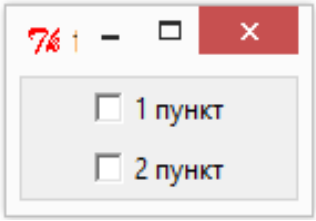
Checkbutton (прапорець) - це віджет, який дозволяє відмітити „галочкою“ певні пункти у вікні. Прапорці не зв’язані між собою, але змінні-константи тут потрібні, щоб знімати дані про стан прапорця. За значенням змінної, що зв’язана з прапорцем (Checkbutton), можна визначити: встановлено прапорець чи ні. Кожен прапорець повинен мати свою прикріплену змінну Tkinter.

Наприклад:

```
from Tkinter import *
root=Tk()
var1=IntVar()
var2=IntVar()
check1=Checkbutton(root,text=u'1 пункт',variable=var1,onvalue=1,offvalue=0)
check2=Checkbutton(root,text=u'2 пункт',variable=var2,onvalue=1,offvalue=0)
check1.pack()
check2.pack()
root.mainloop()
```

З допомогою опції onvalue встановлюється значення, якого набуває зв’язана з прапорцем змінна, коли прапорець включений (обраний).

З допомогою опції offvalue встановлюється значення, якого набуває зв’язана з прапорцем змінна, коли прапорець виключений (необраний).



.

Завдання для самостійного виконання

І рівень

1. Встановити відповідність між класом та його назвою.

А. Radiobutton	1. Рамка
Б. Checkbutton	2. Перемикач
В. Frame	3. Прапорець
Г. Listbox	4. Список

2. Описати значення параметрів у вказаних рядках програми:

```
listbox1=Listbox(root,height=5,width=15,selectmode=EXTENDED)
```

root _____

height=5 _____

width=15 _____

selectmode= EXTENDED _____

```
frame1=Frame(root,bg='green',bd=5)
```

root _____

bg='green' _____

bd=5 _____

```
check1=Checkbutton(root,text=u'1 пункт',variable=var1,onvalue=1,offvalue=0)
```

root _____

text=u'1 пункт' _____

variable=var1 _____

onvalue=1 _____

offvalue=0 _____

```
rbutton3 = Radiobutton (root, text='Варіант 3', variable=v, value=3)
```

root _____

text='Варіант 3' _____

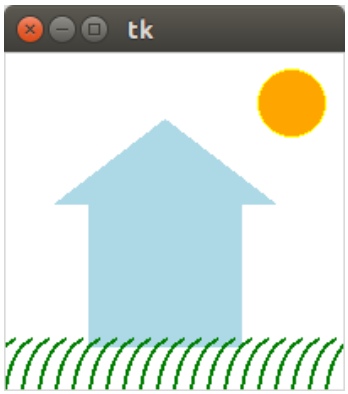
variable=v _____

value=3 _____

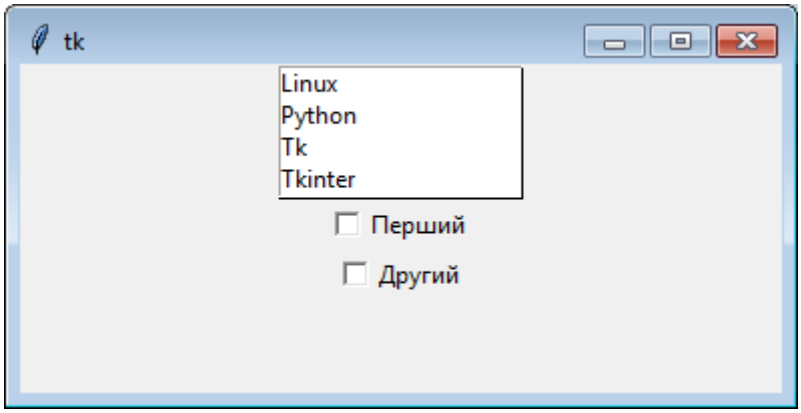
3. Створити вказані у посібнику екранні форми.
4. Створити програму для зображення основних геометричних фігур (задати властивості ліній та кольори на свій смак)

II рівень

5. Написати програму для створення списку свого класу.
6. Написати програму для вибору із переліку навчальних предметів своїх улюблених.
7. Внести зміни в програму із розділу § 6. Екранні форми: перемикачі. Клас Radiobutton , щоб отримати макет керованого світлофора.
8. Створити на полотні подібне зображення:

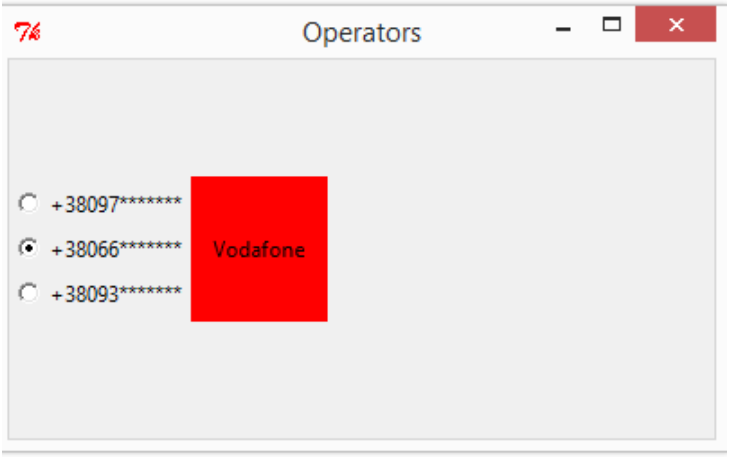


9. Створити програму з елементами керування, щоб отримати таке вікно.

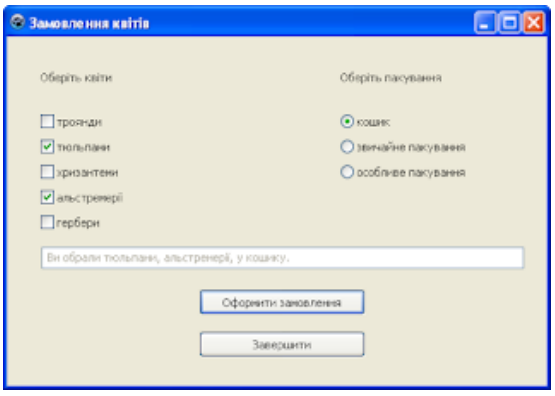


III рівень

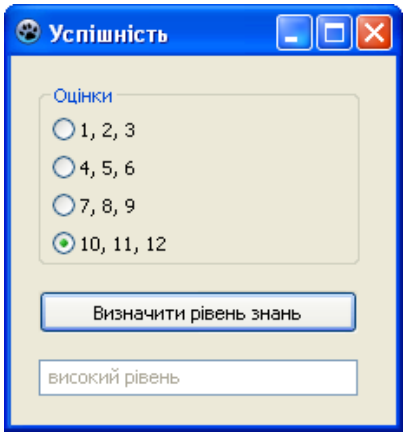
10. Розробіть проект Мобільні оператори, за яким виводиться повідомлення з назвою оператора після вибору певного шаблону номера телефону. Орієнтовний зразок:



11. Розробіть проект Замовлення квітів для оформлення електронного замовлення квітів за зразком



12. Розробіть проект Оцінки, за яким у текстове поле виводиться опис досягнутого рівня на основі шкільної оцінки, яка обирається з елемента управління (оберіть самостійно – прапорці, перемикачі). Орієнтовний зразок інтерфейсу проекту:



§ 8. Математичні рівняння і задачі.

1. **Знайти площу і периметр прямокутного трикутника за двома заданими катетам.**

Підказка.

Площа прямокутного трикутника дорівнює половині площі прямокутника, сторони якого рівні довжині катетів. Периметр же знаходиться шляхом складання довжин всіх сторін трикутника.

Оскільки відомі тільки катети, то гіпотенузу можна знайти за теоремою Піфагора: $c^2 = a^2 + b^2$.

Щоб витягти квадратний корінь в Python можна скористатися функцією sqrt () з модуля math.

2. **Знайти значення функції y = f (x), якщо:**

y = x - 0.5, при x> 0;

y = 0, при x = 0;

y = | x |, при x <0.

Написати програму, що визначає значення y за поданим значенням x.

Підказка.

Щоб подати модуль числа в Python можна скористатися функцією abs () з модуля math.

3. **Знайти корені квадратного рівняння.**

Підказка.

Квадратне рівняння має вигляд $ax^2 + bx + c = 0$. При його вирішенні спочатку обчислюють дискримінант за формулою $D = b^2 - 4ac$. Якщо D>0, то квадратне рівняння має два кореня; якщо D=0, то 1 корінь; і якщо D<0, то роблять висновок, що коріння немає.

4. **Знайти площу прямокутника, трикутника або кола. Залежно від того, що ви обрали, обчислити площу або прямокутника, або трикутника, або кола. Якщо обрані прямокутник або трикутник, то треба запросити довжини сторін, якщо коло, то його радіус.**

Підказка.

Площа прямокутника S=ab

Площа трикутника обчислюється за формулою Герона:

$$S = \sqrt{p(p-a)(p-b)(p-c)}, \text{ де } p = \frac{a+b+c}{2}$$ — половина периметру трикутника.

Площа круга $S= \pi R^2$

5. **Визначити існування трикутника за трьома сторонами**

Підказка.

У трикутника сума будь-яких двох сторін повинна бути більше третьою. Інакше дві сторони просто "ляжуть" на третю і трикутника не вийде.

Користувач вводить довжини трьох сторін. Програма повинна визначати, чи може існувати трикутник при таких довжинах.

6. **Вивести степені натуральних чисел, які не перевищують задане число n. Користувач задає показник степеня та число n.**

Приклад виконання програми:

Показник степеня: 3

Найбільше число: 555

1 8 27 64 125 216 343 512

Останнє число, що піднімається до степеня: 8

7. **Вивести таблицю значень заданої функції на відрізку і з кроком, які вводить користувач. Нехай фунція буде такою: $y = -3x^2 - 4x + 20$.**

Підказка.

- ❖ Запросити у користувача точки початку і кінця відрізка, а також крок.
- ❖ Якщо значення точки початку відрізка більше значення точки кінця, то поміняти значення.
- ❖ Поки значення першої точки не досягне другої: обчислити значення функції, вивести на екран поточні значення x і y, збільшити значення першої точки на крок.

§ 1. Поняття одновимірного масиву

Масивом (таблицею) називають упорядковану сукупність елементів одного типу.

Це значить, що змінна-масив містить не одне значення, а одразу декілька. Їх порядок чітко визначений, отже у кожного значення в масиві є порядковий номер - індекс, за яким можна звернутися до окремого елемента масиву, щоб прочитати або записати значення. Число індексів характеризує розмірність масиву. Кожен індекс змінюється в деякому діапазоні [a, b]. Індекси задають цілочисельним типом.

Кожен елемент масиву є пронумерований і нумерація їхня починається з нуля – тобто перший елемент в масиві має номер 0, другий – 1, і т.д. Ми можемо видаляти елементи з масиву, також додавати нові елементи.

Для визначення масиву його елементи беруться в квадратні дужки.

list = ['blue', 'white', 'black', 'red', 'green', 'yellow']

list[0] = = 'blue'

list[1] = = 'white'

list[4] = = 'green'

list[5] = = 'yellow'

Найцікавішим є те, що до масиву можна застосовувати різні вбудовані функції, призначені для роботи з ними (Таб.6).

Функція	Значення	Приклад
len	Визначає довжину масиву	A=[2, 6, 9] Len (A) = = 3
append	Додає новий елемент	A=[2, 6, 9] A.append (5) [2, 6, 9, 5]
range	Проходить по індексах	range(7) [0, 1, 2, 3, 4, 5, 6]

Таб.6

§ 2. Алгоритми опрацювання масиву

Всі задачі на опрацювання масиву зводяться до кількох типів:

- ❖ знайти суму певних елементів
- ❖ знайти добуток певних елементів
- ❖ знайти елементи, що відповідають певній умові
- ❖ знайти найбільший чи найменший елемент
- ❖ відсортувати елементи масиву

Оскільки масив – це сукупність елементів, то для роботи з масивом слід:

1) ініалізувати масив (певний чином ввести його в програму);
2) визначити умову задачі для обробки елементів масиву;
3) виконати обробку елементів масиву за умовою задачі;
4) вивести результат.

Розглянемо детальніше кожен пункт вказаного вище алгоритму на прикладі кількох задач.

Обчислити суму елементів масиву A, що містить 10 елементів дійсного типу.

1)Ініціалізувати масив	I спосіб. Присвоєння. В комірку з іменем **A** записуємо 10 чисел	A=[11, 25, -13, 66, 36, 90, 18, 17, 20, -51]
2) Визначити умову задачі для обробки елементів масиву;	Знайти суму всіх введених чисел. Для цього треба зарезервувати комірку **S**, куди поміщатимемо по черзі всі елементи та додаватимемо їх. Для того, щоб значення комірки не впливало на результат додавання, запишемо в комірку **S** число 0.	**S=0**
3) Виконати обробку елементів масиву за умовою задачі;	До попередньої суми **S** додаємо значення кожного елементу **A[i]** масиву	**S=S+A[i]**

4) Вивести результат.	Оскільки в задачі вимагалось обчислити суму елементів масиву, то результатом буде значення змінної **S**	print **S**

Перевіримо виконання алгоритму з допомогою таблиці:

Лічильник	Сума	Елемент масиву
	S	
i	0 – початкове значення	**A[i]**
0	0+11= = 11	A[0]= =11
1	11+25= = 36	A[1]= =25
2	36+(-13)= = 23	A[2]= =-13
3	23+66= = 89	A[3]= =66
4	89+36= = 125	A[4]= =36
5	125+90= = 215	A[5]= =90
6	215+18= =233	A[6]= =18
7	233+17= =250	A[7]= =17
8	250+20= = 270	A[8]= =20
9	270+(-51)= =219	A[9]= =-51
	S= = 219	

Оскільки потрібно опрацювати кожен елемент масиву, то дії виконуються обов'язково в циклі, який має наступний синтаксис:

for i in range(len(A)):

лічильник і проходить значення від 0 до 9, тому що:

- len(A)=10
- range (10) = [0, 1, 2, 3, 4, 5, 6, 7, 8, 9]

Отже, лічильник **i** по черзі приймає значення від 0 до 9.

Програма для розв'язку задачі може мати такий вигляд (пам'ятаємо про відступи):

```
A=[11,25,-13,66,36,90,18,17,20,-51]
S=0
```

```
for i in range(len(A)):
    S=S+A[i]
print S
```

§ 3. Знаходження елементів, що задовольняють задані умови

Вивести всі елементи масиву з парними індексами.

1) Ініціалізувати масив	II спосіб. З клавіатури. В комірку з іменем **n** записуємо бажану кількість елементів	n=int(input("Rozmir masiva: "))
	В комірку з іменем **A** записуємо порожній масив	A = []
	Вводимо з клавіатури в порожній масив A елементи, записуючи їх в кінець масиву по черзі, поки їх не стане **n**	while n > 0: A.append(int(input())) n=n- 1
2) Визначити умову задачі для обробки елементів масиву;	Знайти всі парні індекси. Умова парного індексу: в остачі від ділення індексу **i** на 2 маємо нуль.	if i%2 = = 0:
3) Виконати обробку елементів масиву за умовою задачі;	Ці етапи можна об'єднати, оскільки елементи з парними числами потрібно відразу вивести на екран.	print A[i]
4) Вивести результат.		

Перевіримо виконання алгоритму з допомогою таблиці:

n=6	Перевірка умови **n > 0**	Заповнення масиву **A[ ]**
6	6>0	A=[11]
5	5>0	A=[11, 32]
4	4>0	A=[11, 32, 5]
3	3>0	A=[11, 32, 5, 0]
2	2>0	A=[11, 32, 5, 0, -3]

1	1>0	A=[11, 32, 5, 0, -3, -15]
0	0>0	Цикл введення закінчено
Лічильник	Перевірка умови	Виведення
0	0%2= =0	11
1	1%2= =1	
2	2%2= =0	5
3	3%2= =1	
4	4%2= =0	-3
5	5%2= =1	

Програма для розв'язку задачі може мати такий вигляд (пам'ятаємо про відступи):

```
n = int(input("Rozmir masiva: "))
A = []
while n > 0:
    A.append(int(input()))
    n=n- 1
for i in range(len(A)):
    if i%2==0:
        print A[i]
```

§ 4. Знаходження підсумкових величин для елементів, що задовольняють задані умови

Обчислити добуток додатних елементів масиву MAS[1..10]. Елементи масиву є довільними числами.

1) Ініціалізувати масив	II спосіб. З клавіатури. В комірку з іменем **n** записуємо вказану кількість елементів	n=10
	В комірку з іменем **MAS** записуємо	**MAS** = []

	порожній масив	
	Вводимо з клавіатури в порожній масив **MAS** елементи, записуючи їх в кінець масиву по черзі, поки їх не стане **n**	while n > 0: MAS.append(int(input())) n=n- 1
2) Визначити умову задачі для обробки елементів масиву;	Додатні елементи більші нуля.	if **MAS** [i]>0:
	Знайти добуток вибраних чисел. Для цього треба зарезервувати комірку **P**, куди поміщатимемо по черзі обрані елементи та перемножатимемо їх. Для того, щоб значення комірки не впливало на результат множення, запишемо в комірку **P** число 1	P=1
3) Виконати обробку елементів масиву за умовою задачі;	До попереднього добутку **P** домножаємо значення кожного відібраного елементу **MAS [i]** масиву	P=P* **MAS** [i]
4) Вивести результат.	Оскільки в задачі вимагалось обчислити добуток додатніх елементів масиву, то результатом буде значення змінної **P**	print P

Перевіримо виконання алгоритму з допомогою таблиці:

n=10	Перевірка умови **n > 0**	Заповнення масиву **A[ ]**
10	10>0	MAS=[11]
9	9>0	MAS=[11, 32]
8	8>0	MAS=[11, 32, 5]
7	7>0	MAS=[11, 32, 5, 0]
6	6>0	MAS=[11, 32, 5, 0, -3]

5	5>0	MAS=[11, 32, 5, 0, -3, -15]
4	4>0	MAS=[11, 32, 5, 0, -3, -15, 0]
3	3>0	MAS=[11, 32, 5, 0, -3, -15, 0, 2]
2	2>0	MAS=[11, 32, 5, 0, -3, -15, 0, 2, 5]
1	1>0	MAS= [11, 32, 5, 0, -3, -15, 0, 2, 5, -3]
0	0>0	Цикл введення закінчено

Лічильник	Елемент масиву		Добуток
			P
i	MAS **[i]**	MAS **[i]>0**	1 – початкове значення
0	MAS[0]= =11	True	1*11= =11
1	MAS [1]= =32	True	11*32= =352
2	MAS [2]= =5	True	352*5= =1760
3	MAS [3]= =0	False	
4	MAS [4]= =-3	False	
5	MAS [5]= =-15	False	
6	MAS [6]= =0	False	
7	MAS [7]= =2	True	1760*2= =3520
8	MAS [8]= =5	True	3520*5= =17600
9	MAS [9]= =-3	False	
			P= = 17600

Програма для розв’язку задачі може мати такий вигляд (пам’ятаємо про відступи):

```
n = 10
MAS = []
while n > 0:
   MAS.append(int(input()))
    n=n- 1
P=1
```

```
for i in range(10):
    if MAS [i]>0:
        P=P*MAS[i]
print P
```

§ 5. Алгоритми впорядкування масиву.

Сортування бульбашкою - найпростіший алгоритм сортування, застосовуваний чисто для навчальних цілей. До плюсів сортування бульбашкою відноситься простота реалізації алгоритму.

Алгоритм сортування бульбашкою зводиться до повторення проходів по елементах сортованого масиву. Прохід по елементах масиву виконує внутрішній цикл. За кожен прохід порівнюються два сусідні елементи, які за потребою міняються місцями. Зовнішній цикл буде працювати доти, поки масив не буде відсортований. Таким чином зовнішній цикл контролює кількість спрацьовувань внутрішнього циклу. Коли при черговому проході за елементами масиву не буде здійснено жодної перестановки, то масив буде вважатися відсортованим.

Алгоритм сортування бульбашкою гарно показано на інтернет-ресурсі (Мал.24):

https://www.youtube.com/watch?v=5JMInXAtnQg

Мал.24

Програма для сортування введеного з клавіатури масиву може мати такий вигляд (пам’ятаємо про відступи). Спробуйте самостійно визначити призначення

кожного рядочка.

```
n=int(input())
F=[]
while n>0:
    F.append(int(input()))
    n=n-1
n=len(F)
for i in range(n):
    for j in range(n-1):
        if F[j]>F[j+1]:
            k=F[j]
            F[j]=F[j+1]
            F[j+1]=k
Print F
```

Для двох заданих масивів: відсортуйте перший — у порядку зростання, другий — у порядку спадання. Утворіть третій як попарну суму відповідних елементів. Виведіть утворений масив.

Проаналізувавши вказану задачі, розіб'ємо її на під задачі.

1. Введення двох масивів.
2. Сортування масивів.
3. Створення третього масиву.

Програма для розв'язку задачі може мати такий вигляд (пам'ятаємо про відступи):

```
n=int(input('vvedit rozmir masiva A   '))
A=[]
B=[]
F=[]
while n>0:
    A.append(int(input()))
```

```
    n=n-1
n=int(input('vvedit rozmir masiva B   '))
while n>0:
    B.append(int(input()))
    n=n-1
n=len(A)
for i in range(n):
    for j in range(n-1):
        if A[j]>A[j+1]:
            k=A[j]
            A[j]=A[j+1]
            A[j+1]=k
for i in range(n):
    for j in range(n-1):
        if B[j]<B[j+1]:
            l=B[j]
            B[j]=B[j+1]
            B[j+1]=l
for i in range(n):
    F.append(A[i]+B[i])
print 'A=', A
print 'B=', B
print 'F=', F
```

Завдання для самостійного виконання

І рівень

1. Запишіть умову: елемент масиву більший 7.
2. Запишіть умову: елемент масиву від’ємний.
3. Запишіть умову: елемент масиву ділиться на 4.
4. Запишіть умову: елемент масиву парний.

5. Запишіть початкове значення суми.
6. Запишіть початкове значення добутку.
7. Запишіть початкове значення кількості елементів.
8. Запишіть оператор визначення довжини масиву К.
9. Запишіть числове значення довжини масиву, у якому записують середню денну температуру протягом тижня.
10. Запишіть оператор присвоєння, який виконує таку дію: третьому елементу масиву С присвоїти значення суми першого і четвертого елементів;
11. Запишіть оператор присвоєння, який виконує таку дію: другому елементу масиву С присвоїти значення його подвоєного добутку.
12. Запишіть умову: перший елемент масиву більший за другий.

ІІ рівень

Написати програми для розв'язування задач:

1. Обчислити добуток елементів масиву
2. Обчислити кількість елементів масиву
3. Обчислити суму елементів масиву, які менші за 13
4. Вивести на екран всі елементи масиву, які діляться на 5
5. Обчислити добуток додатніх елементів масиву
6. Обчислити кількість додатніх елементів масиву
7. Вивести на екран всі нульові елементи масиву.
8. Знайти кількість від'ємних елементів масиву.
9. Додатні елементи масиву збільшіть удвічі.
10. Від'ємні елементи масиву зменшіть на одиницю.
11. Виведіть на екран ті елементи масиву, які менші за 4.
12. Вивести на екран номери від'ємних елементів масиву.
13. Створити масив, у якому кожен елемент рівний своєму номеру, збільшеному на 2.
14. Вивести на екран всі парні елементи масиву.

15. Знайти кількість непарних елементів масиву.
16. Знайти суму квадратів елементів масиву.
17. Знайти квадрат суми додатних елементів масиву.

III рівень

1. Обчислити середнє значення набору числових даних
2. Елементи табличної величини, яка містить 10 цілих чисел, змінити їх квадратами
3. Знайти суму додатних і кількість від'ємних елементів лінійної таблиці дійсного типу
4. Замінити найменший елемент масиву нулем.
5. Визначити номер максимального елемента масиву.
6. Занести елементи послідовності sin3i, i=1,2,…,n до масиву та вивести їх на екран.
7. Записати умову задачі, яка розв'язується таким кодом:

```
z = input("Rozmir masiva: "))
arr = []
while z > 0:
   arr.append(input())
   z=z-1
maxim=arr[0]
minim=arr[0]
for i in range(len(arr)):
   if arr[i]>maxim:
      maxim=arr[i]
for i in range(len(arr)):
   if minim>arr[i]:
      minim=arr[i]
print maxim+minim
```

Завдання для розумників

1. Створіть програму «Продаж товару», у якій для введеної кількості проданих одиниць товару, яка фіксується кожну годину за зміну (8 год), знаходять підсумкове значення, яке виводиться на екран.
2. Створіть програму «Надбавка», у якому заробітна плата 10 працівників деякого підприємства, подана дійсними величинами у лінійній таблиці, збільшується на 15%.
3. У деякій таблиці міститься значення температури повітря кожного дня тижня. Створіть програму для визначення середньої температури повітря за весь тиждень. З'ясуйте, скільки разів на тиждень середня денна температура була вище нуля.

§ 6. Математичні рівняння і задачі.

1. Вивести рівняння прямої, що проходить через задані точки

Підказка.

Рівняння прямої на координатної площині має наступний вигляд: y=kx+b. Якщо відомі координати двох точок, що лежать на цій прямій, то можна, розв'язавши систему рівнянь, визначити значення коефіцієнтів k і b. Таким чином виводиться рівняння конкретної прямий, наприклад, y = 3x - 1.

Розв'язуємо систему рівнянь:

y1 = kx1 + b

y2 = kx2 + b

b = y2 - kx2

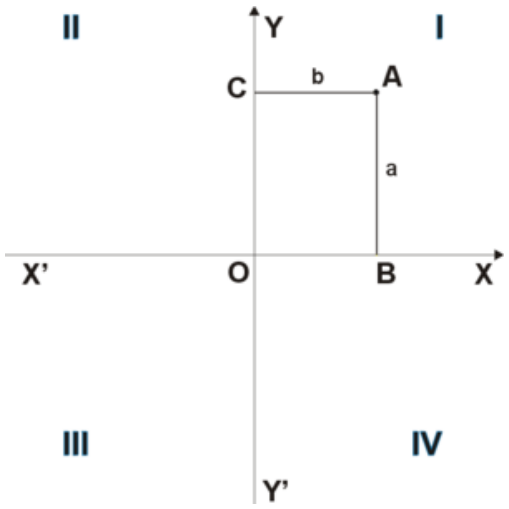
y1 = kx1 + y2 - kx2

k = (y1 - y2) / (x1 - x2)

2. За координатами визначити, в якій чверті знаходиться точка

Підказка.

У прямокутній системі координат на площині виділяють чверті:



I-й чверті відповідають точки, які мають обидві (x і y) позитивні координати.

II-а чверть: x <0, y> 0.

III-я чверть: x <0, y <0.

IV-а чверть: x> 0, y <0.

3. Чи належить точка колу? Визначити, чи належить точка з координатами (x; y) колу радіусом R з центром на початку координат.

Підказка.

Користувач вводить координати точки і радіус кола.

Якщо вибрати точку на координатній площині, то можна побачити, що проекції її координат на осі x і y є катетами прямокутного трикутника. А гіпотенуза цього прямокутного трикутника показує відстань від початку координат до точки. Таким чином, якщо довжина гіпотенузи буде менше радіуса кола, то точка буде належати колі; інакше вона буде знаходиться за його межами.

Довжину гіпотенузи можна обчислити по теоремі Піфагора: квадрат гіпотенузи дорівнює сумі квадратів катетів.

4. Знайти суму n елементів ряду 1, -0.5, 0.25, -0.125, ...

Підказка.

В даному випадку бачимо, що кожен наступний елемент в 2 рази менший попереднього по модулю, але взятий з протилежним знаком. Далі слід знайти арифметична дія, яке з попереднього елемента дає наступний. Тут, наприклад, треба попередній елемент ділити на -2.

5. Знайти суму і кількість елементів послідовності, які по модулю більше 0.001.

Послідовність: 1/2 - 2/4 + 3/8 - 4/16 + ... - ...

Підказка.

В даній послідовності кожний наступний елемент відрізняється від попереднього:

- знаком,
- чисельник збільшений на 1,
- знаменник збільшений в 2 рази.

Значить кожен наступний елемент ряду обчислюється з поточного за формулою - (a + 1) / (b * 2), де a і b - чисельник і знаменник дробу числа ряду.

6. Функція, що обчислює площі різних геометричних фігур

Припустимо, потрібно написати функцію, яка може обчислювати площу кола, прямокутника і трикутника. Площа якої фігури необхідно обчислити, має визначатися переданими аргументами. Так для обчислення площі круга досить знати лише його радіус. Щоб обчислити площу прямокутника, потрібно вже два числа - його довжина і ширина. Для визначення площі трикутника, наприклад, за формулою Герона необхідні довжини всіх трьох сторін, тобто в функцію треба передати три числа.

Щоб у користувача не питати, площа якої фігури він хоче дізнатися, можна запитувати введення даних одним рядком і саме її передавати в функцію. У функції рядок треба перетворити в список чисел і в залежності від їх кількості провести обчислення по тій чи іншій формулі.

Список використаних джерел

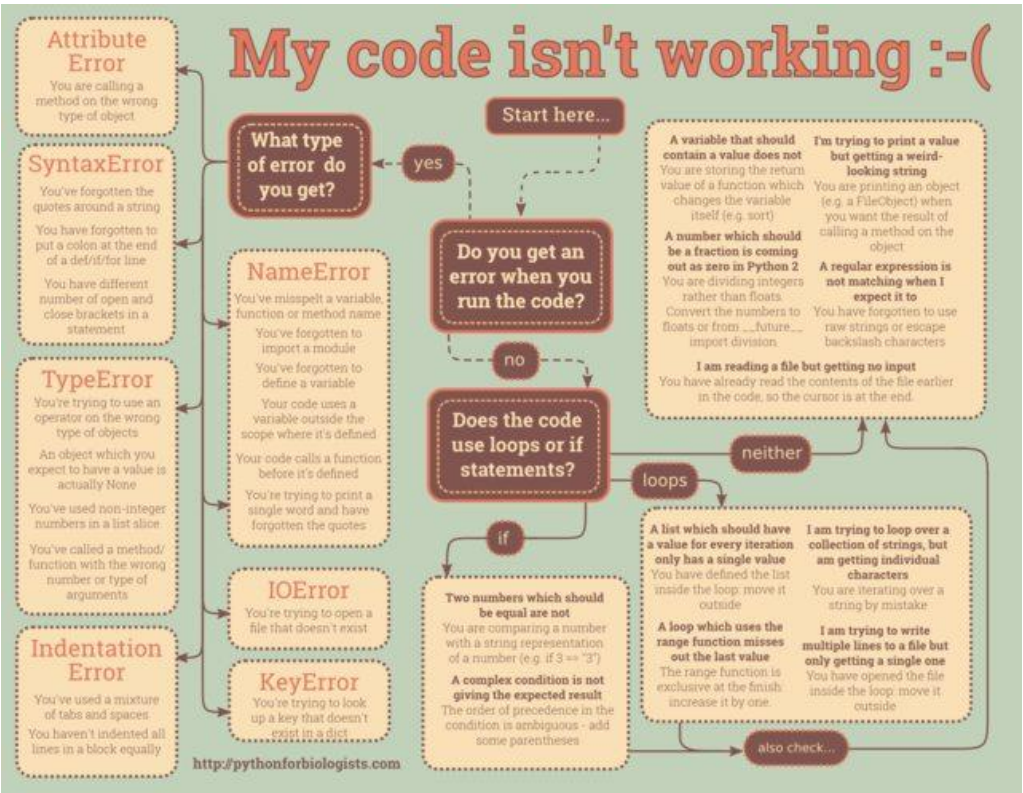
1. «Мовчазний учитель» он-лайн тренажер [Електронний ресурс]: - Режим доступу: http://silentteacher.toxicode.fr/hourofcode
2. Python. Обучение программированию [Електронний ресурс]: - Режим доступу: https://younglinux.info/python
3. Python. Tk_canvas [Електронний ресурс]: - Режим доступу: https://www.tutorialspoint.com/python/tk_canvas.htm
4. Безкоштовні масові он-лайн курси [Електронний ресурс]: - Режим доступу:http://prometheus.org.ua/
5. Бібліотека Tkinter мови Python [Електронний ресурс]: - Режим доступу: http://www.kievoit.ippo.kubg.edu.ua/kievoit/2016/tkinter/index.html
6. Блог програміста [Електронний ресурс]: - Режим доступу: http://www.vitaliypodoba.com
7. Вікіпедія [Електронний ресурс]: - Режим доступу: uk.wikipedia.org
8. Дистанційна підтримка освіти школярів [Електронний ресурс]: - Режим доступу: http://disted.edu.vn.ua/
9. Довідкове керівництво. [Електронний ресурс]: - Режим доступу: Wing IDE 101
10. Клас (програмування) [Електронний ресурс]: - Режим доступу: https://uk.wikipedia.org/wiki/
11. Класи_та_ітератори [Електронний ресурс]: - Режим доступу: https://uk.wikibooks.org/wiki/Пориньте_у_Python_3/
12. Курс_по_библиотеке_Tkinter_языка_Python [Електронний ресурс]: - Режим доступу: https://ru.wikiversity.org/wiki/
13. Об'єктно-орієнтоване програмування [Електронний ресурс]: - Режим доступу: https://learn.ztu.edu.ua/mod/page/view.php?id=7254
14. Об'єктно-орієнтоване програмування [Електронний ресурс]: - Режим доступу: http://programer.in.ua/index.php/prohramuvannia/prohramuvannia-na-

c/127-urok-12-poniattia-ob-iektno-oriientovanoho-prohramuvannia-oop-klasy-i-ob-iekty

15. Обробники цих подій мовою Python [Електронний ресурс]: - Режим доступу: http://www.kievoit.ippo.kubg.edu.ua/kievoit/2016/48_Python/index.html
16. Програмування по-українськи [Електронний ресурс]: - Режим доступу: http://programming.in.ua/
17. Tkinter-canvas-python [Електронний ресурс]: - Режим доступу: http://poligraph.blogspot.com/2010/01/tkinter-canvas-python.html

Додаток

Основні помилки при кодуванні



My code isn't working	Мій код не працює
Do you get an error when you run the code?	Ви отримуєте повідомлення про помилку під час запуску коду?
What type of error do you get?	Який тип помилки ви отримуєте?
Does the code use loops or if statements?	Чи використовуються дужки або if в коді?
Name error You've misspelt a variable, function of method name	Помилка імені Ви допустили орфографічну помилку при написанні змінної, функції чи модуля
Name error You've forgotten to import a module	Помилка імені Ви забули приєднати модуль
Name error You've forgotten to define a variable	Помилка імені Ви забули визначити змінну

Name error Your code uses a variable outside the scope where it's defined	Помилка імені Ваш код використовує змінну поза областю, де вона визначена
Name error Your code calls a function before it's defined	Помилка імені Ваш код викликає функцію перш, ніж вона описана
You're trying to print a single word and have forgotten the quotes	Ви намагаєтеся надрукувати слово і забули лапки
IOError You're trying to open a file that doesn't exist	Помилки введення-виведення Ви намагаєтеся відкрити файл, який не існує
Key error You're trying to look up a key that doesn't exist in a dict	Помилка ключа Ви намагаєтеся шукати ключ, який не існує в Словнику
Attribute error You are calling a method on the wrong type of object	Помилки атрибуту Ви викликаєте дію на невідповідний тип об'єкта
Syntax error You've forgotten the quotes around the string	Синтаксична помилка Ви забули лапки навколо рядка
Syntax error You have forgotten to put a colon at the end of a def / if / for line	Синтаксична помилка Ви забули поставити двокрапку в кінці лінії def / if / for
Syntax error You have different number of open and close brackets in a statement	Синтаксична помилка У вас не співпадає кількість відкритих і закритих дужок в коді
Type Error You're trying to use an operator on the wrong type of object	Помилки типів Ви намагаєтеся використовувати оператор на невідповідний тип об'єкта

Type Error An object which you expect to have a value is actually None	Помилки типів Очікуване значення змінної є порожнім
Type Error You've used non-integer numbers in a list	Помилки типів Ви використовували нецілі числа в списку
Type Error You've called a method / function which the wrong numbers or type of arguments	Помилки типів Ви викликали модуль / функцію, використовуючи невідповідні числа або типи аргументів
Indentation Error You've used of mixture of tabs and spaces	Помилки відступів Ви використали одночасно табуляцію та пробіли
Indentation Error You haven't indent on lines in a block equally	Помилки відступів Ви не однаково виставили відступи рядків в блоці
Two numbers which should be equal are not You are comparing a number with a string representation of a number (if 3==’3’)	Два числа, які повинні бути рівні, не є рівними Ви порівнюєте число з символом (if 3==’3’)
A complex condition is not giving the expected result The order of precedence in the condition is ambiguous - add some parentheses	Складена умова не дає очікуваного результату Порядок дій в умовах неоднозначний – додай круглі дужки
A variable that should contain a value does not	Змінна, яка повинна містити значення, порожня

You are storing the return value or a function which changes the variable itself	Ви використали функцію, яка викликає саму себе (рекурсія)
A number which should be a fraction is coming out as zero in Python 2 You are dividing integers rather than float. Convert the numbers to floats or from ... future ... import division	Число, яке повинно бути дробовим виходить як нуль в Python 2 Ви розділили цілі числа, а не дійсні. Перетворюючи числа в дійсні або навпаки ... перед тим ... підключіть модуль
I am reading a file but getting no input You have already read the contents of the file earlier in the code, so the cursor is at the end	Я зчитую файл, але не отримуючи ніякого введення Ви вже зчитали вміст файлу раніше в коді, так що курсор знаходиться в кінці
I'm trying to print a value but getting a weird-looking string You are printing an object when you want the result of calling a method on the object	Я намагаюся надрукувати значення, але отримую дивний вигляд рядка Ви друкуєте в об'єкт, коли ви хочете викликати результат
A regular expression is not matching when I expect it to You have forgotten to use raw strings of escape backslash characters	вираз не відповідає, коли я очікую виведення Ви забули використати рядок символів чи повернути косі риски