С.П. Негода Збірник навчальних практичних робіт з програмування мовою Pascal

Автор: Негода Сергій Петрович

1. Лінійні алгоритми мовою Pascal

Практична робота 1.

Лінійні алгоритми мовою Pascal

Завдання 1.(3 бали). Скласти і реалізувати алгоритм в програмному середовищі, який визначає із довільної кількості прямих загального положення число точок перетину цих прямих. Значення кількості прямих вводяться з клавіатури; і виводить на екран кількість точок перетину цих прямих. Використовувати в алгоритмі більше однієї цілої змінної не можна.

Алгоритм мовою Pascal

```
Program linearPunkt;   {алгоритм знаходження кількості точок перетину прямих на площині}

var                    { оголошується опис змінних величин, які використовує алгоритм}

k: integer;            {оголошується  ціла змінна у даному  алгоритмі}

begin                                  {початок виконання дій алгоритму}

write('k='); readln(k);      { на екрані запит на введення з клавіатури цілого числа: у}

 write('кількість прямих k='); writeln(k,'  ');       { перевірка початкового: k}

k:=(k-1)*k div 2;              {обчис-ня  кіл-сті точок перетину за  форм-лою:0,5(k-1)k}

writeln;  write('кількість точок перетину прямих k=');     {повідомлення про виведення результату}

writeln(k,'  ');               {виведення результату обчислення алгоритму}

writeln;    end.                {закінчення дій алгоритму}
```

Протестувати правильну роботу цього алгоритму для цілих чисел: 1; 2; 4; 5; 0; 100; 901; 9000; 101010.

Завдання 2.(3 бали). Скласти і реалізувати алгоритм в програмному середовищі, який виконує взаємний обмін числовими значеннями двох цілих змінних; і виводить на екран два початкових значення змінних і два кінцеві значення цих змінних. Використовувати в алгоритмі більше двох цілих змінних не можна.

Алгоритм мовою Pascal

```
program SuperposiziaNumer;   {алгоритм обміну числовими значеннями двох змінних}

var                  {оголошується про опис змінних величин, які використовує алгоритм}

x,y: integer;                    {оголошуються дві цілі змінні у даному алгоритмі}

begin                              {оголошується про початок алгоритмічних дій}

write('x=');              {на екрані запит на введення з клавіатури цілого числа: x}

readln(x);              {зчитування числового значення і внесення його в змінну x}

write('y=');              { на екрані запит на введення з клавіатури цілого числа: y}

readln(y);              {зчитування числового значення і внесення його в змінну y}

 write('початкове значення x='); writeln(x,'  ');          { виведення на екран початкового: x}

 write('початкове значення y='); write(y,'  ');              { виведення на екран початкового: y}

x:=x+y; y:=x-y; x:=x-y;    {взаємний обмін числами за допомогою арифметичних дій «+»та «-».}

writeln;              {поставити курсор на екрані з нового рядка}

 write('кінцеве значення x=');    {виведення на екран повідомлення}

writeln(x,'    ');        {виведення на екрані числового значення змінної x}

 write('кінцеве значення y=');        {виведення на екран повідомлення}

write(y,'    ');                  {виведення на екрані значення значення змінної y}

 writeln;                  {поставити курсор на екрані з нового рядка}

end.                              {закінчення дій алгоритму}
```

Протестувати правильну роботу цього алгоритму для пари цілих чисел: -9 і 17; 0 і 7; 100 і; 250; -900 і - 1000. Виконайте цей алгоритм для дійсного типу числових

величин.

Завдання 3. **(3 бали).** Самостійно скласти і реалізувати алгоритм в програмному середовищі, який визначає із довільної кількості прямих загального положення на площині знаходить кількість утворених частин площини, при умові, що кожна точка перетину прямих утворена не більше, ніж двома прямими. Значення кількості прямих вводиться з клавіатури; і результат виводить на екран, це кількість частин площини. Використовувати в алгоритмі більше однієї цілої змінної не можна. Формула кількості частин на площині після перетину n непаралельних прямих має вигляд: k(n) = **(n+1)n/2 +1**, де n - натуральне число, кількість прямих.

Протестувати правильну роботу цього алгоритму для цілих чисел: 1; 2; 3; 4; 5; 6; 7; 3059; 121314.

Завдання 4. (3 бали). Самостійно скласти і реалізувати алгоритм в програмному середовищі, що за заданими n точками на площині, знаходить кількість відрізків між цими усіма точками. Протестувати правильну роботу цього алгоритму для цілих чисел: 1; 2; 3; 4; 5; 6; 7; 2099; 521314. Формула кіль-сті відрізків: k(n) = **(n-1)n/2** , де n - натуральне число, кількість точок.

Практична робота 2.

Лінійні алгоритми мовою Pascal

Завдання 1. Скласти і реалізувати алгоритм для знаходження кількості днів, за яку виконають сумісну роботу два програміста, якщо такий об'єм роботи перший програміст самостійно виконує за k днів, а другий програміст самостійно виконує за m днів.

```
program  Pobota_1;                  {назва    алгоритму}

var  k,m: real;                     {оголошення  змінних величин: k,m – це дійсні числа}

begin                                                    { початок   виконання
алгоритму}

  writeln( 'k='); readln(k); writeln( 'm='); readln(m);    { введення двох чисел}

   k:=(k*m)/(k+m);                                        { обчислення за формулою}

writeln('Разом виконають за ', k, ' днів');              { виведення результату}

end.                                                     {закінчення алгоритму}
```

Протестуйте алгоритм для таких значень kim: а)12і 8; б)3,2і 2,4; в)6,5і 2,6; г)42 і36; д)40,30 і 40,45; е)20 і 16; є)80 і 84.

Завдання 2. Скласти і реалізувати алгоритм для знаходження справжньої маси монети,

якщо її зважували на бракованих терезах з нерівними плечами, і при викладенні гирьок на першій чашечці, то маса монети на протилежній чашечці становила k грам, а при викладанні гирьок на другій чашечці терезів маса монети на протилежній чашечці становила m грам.

```
program  Pobota_2;                {назва    алгоритму}

var  k,m: real;                   {оголошення  змінних величин: k, m – це дійсні числа}

begin                                                  { початок  виконання  алгоритму}

  writeln( 'k='); readln(k); writeln( 'm='); readln(m);    { введення двох чисел}

   k:=sqrt(k*m);                    { обчислення за функцією квадратного кореня}

writeln('Справжня маса монети: ', k, '  грам);              { виведення результату}

end.                                                    {закінчення алгоритму}
```

Протестуйте алгоритм для таких значень k і m: а)12 і 8; б)3,2 і 2,4; в)6,5 і 2,6; г)42 і36; д)40,30 і 40,45; е)20 і 16; є)80 і 84.

Завдання 3. Є три гаманці: татчин, мамчин, сина. У татовому гаманці: **k** грн, у маминому гаманці **m** грн. Бабуся запитала внучка, яку б ти хотів мати суму грошей у своєму гаманці і запропонувала чотири можливі варіанти: 1) середнє арифметичне грошей у двох гаманцях, що мають **k+1000** грн і **m-1000** грн відповідно; 2) середнє геометричне грошей, що у двох гаманцях, котрі мають **k-2000** грн і **m+2000** грн відповідно; 3)середнє квадратичне грошей у двох гаманцях, котрі мають **k-3000** грн і **m+3000** грн відповідно; 4) середнє гармонійне грошей у двох гаманцях, котрі мають **k+4000** грн і **m-4000** грн відповідно. Для внучка бабусі скласти і реалізувати алгоритм для впорядкування від найбільшого до найменшого названих бабусею чотирьох грошових величин.

```
program  Pobota_3;                {назва    алгоритму}

var  k,m,n: real;                  {оголошення  змінних величин: k, m, n – це дійсні числа}

begin                                  { початок  виконання  алгоритму}

  writeln( 'k='); readln(k); writeln( 'm='); readln(m);    { введення двох чисел}

   n:=sqrt(((k-3000)*(k-3000)+(m+3000)*(m+3000))*0.5);   {обчислення за фор-лою
серед-ого квадр-ного}

writeln('Середнє квадратичне: ', n, '  грн ');              { виведення результату}

   n:= ((k+1000)+(m-1000))*0.5;   {обчислення за фор-лою серед-ого арифметичного}

writeln('Середнє арифметичне: ', n, '  грн ');              { виведення результату}

   n:=sqrt((k-2000)*(m+2000));   {обчислення за фор-лою серед-ого геометр-ного}

writeln('Середнє геометричне: ', n, '  грн ');              { виведення результату}
```

n:=2*((k+4000)*(m-4000))/ ((k+4000)+(m-4000)); {обчислення серед-ого гарм-ного}

writeln('Середнє гармонійне: ', n, ' грн '); { виведення результату}

end. {закінчення алгоритму}

Протестуйте алгоритм для таких значень k і m: а)10000 і 80000; б)300000 і 400000; в)600500 і 900600.

–

Практична робота 3.

Лінійні алгоритми мовоюPascal

Завдання 1(2 бали). Скласти і реалізувати алгоритм в програмному середовищі, який визначає із двох довільних дійсних чисел **найменше число та найбільше число**. Значення двох дійсних змінних вводяться зклавіатури; і виводить на екран найменше значення із двох чисел та найбільше значення із двох чисел. Використовувати в алгоритмі більше трьох дійсних змінних не можна.

Алгоритм мовою Pascal

program MAxMinNumer; {алгоритм знаход-ня MAX(x;y) та MIN(x;y)}

var {оголошується опис змінних величин, які використовує алгоритм}

x,y,z: real; {оголошуються три дійсні змінні величини **x,y,z** у даному алгоритмі}

begin {оголошується початок алгоритмічних дій}

write('x='); {на екрані запит на введення з клавіатури цілого числа: x}

readln(x); {зчитування числового значення і внесення його в змінну x}

write('y='); { на екрані запит на введення з клавіатури цілого числа: y}

readln(y); {зчитування числового значення і внесення його в змінну y}

write('початкове значення x='); writeln(x,' '); { виведення на екран початкового: x}

write('початкове значення y='); write(y,' '); { виведення на екран початкового: y}

z:=abs(x-y); x:=0.5*(x+y-z); y:=x+z; **{**знаходження MAX(x;y) та MIN(x;y) }

writeln; {поставити курсор на екрані з нового рядка}

write('найменше значення x='); {виведення на екран повідомлення}

writeln(x,' '); {виведення на екран найменшого числового значення змінної x}

write('найбільше значення y='); {виведення на екран повідомлення}

write(y,' '); {виведення на екрані найбільшого числового значення змінної y}

writeln; {поставити курсор на екрані з нового рядка}

end. {закінчення дій алгоритму}

Протестувати правильну роботу цього алгоритму для двійок цілих чисел: **-9.9 і -1.7;** 0.89 і -8.93; -2 і 0; -1.4 і 1.7; 10.033 і 25.902; 900 і - 1000.

Завдання 2. (2 бали). Самостійно скласти і реалізувати алгоритм в програмному середовищі, що за заданим цілим числом n знаходить суму **перших парних натуральних** чисел, що не перевищують 2n. Протестувати правильну роботу цього алгоритму для цілих чисел: 1; 2; 3; 4; 25; 36; 57; 2022; 521314. Формула суми парних чисел: 2+4+6+..+ 2n = $n*(n+1)$, де n - натуральне число.

Завдання 3. (2 бали). Самостійно скласти і реалізувати алгоритм в програмному середовищі, що за заданим цілим числом n знаходить суму **перших непарних натуральних чисел**, що не перевищують 2n-1. Протестувати правильну роботу цього алгоритму для цілих чисел: 1; 2; 3; 4; 25; 46; 77; 3033; 521314. Формула суми непарних чисел: 1+3+5+...+ (2n-1) = $n*n=n^2$, де n - натуральне число.

Завдання 4. (2 бали). Самостійно скласти і реалізувати алгоритм в програмному середовищі, що за заданим цілим числом n знаходить суму **квадратів натуральних чисел**, що не перевищують n^2. Протестувати правильну роботу цього алгоритму для цілих чисел: 1; 2; 3; 4; 5; 16; 47; 4044; 521314. Формула суми **квадратів** чисел: $1^2+2^2+...+ n^2=$ **n*(n+1)*(2n+1)/6**, (div) де n - натуральне число.

Завдання 5. (2 бали). Самостійно скласти і реалізувати алгоритм в програмному середовищі, що за заданим цілим числом n знаходить суму **кубів натуральних чисел**, що не перевищують n^3. Протестувати правильну роботу цього алгоритму для цілих чисел: 1; 2; 3; 4; 5; 15; 27; 505; 521314. Формула суми **кубів** чисел: $1^3+2^3+3^3+4^3+...+ n^3=$ $n^2*(n+1)^2/4$,(div) де n - натуральне число.

Завдання 6. (2 бали). Самостійно скласти і реалізувати алгоритм в програмному середовищі, що за заданим цілим числом n знаходить суму **четвертих степенів натуральних чисел**, що не перевищують n^4. Протестувати правильну роботу цього алгоритму для цілих чисел: 1; 2; 3; 4; 6; 16; 26; 66; 521314. Формула суми **четвертих степенів** чисел: $1^4+2^4+3^4+...+ n^4=(6n^5+15n^4+10n^3- n)/30$,(div) де n - натуральне число.

Завдання 7.(2 балів). Самостійно скласти і реалізувати алгоритм в програмному середовищі, що за заданим цілим числом n знаходить найменше число та найбільше число для значень $(1-n)*(n^2-5n+6)$ та $(1-n)*(n^2-6n+5)$ **. Цілі** змінні вводяться з клавіатури; результат виводить на екран найменше значення із двох чисел та найбільше значення із двох чисел. Використовувати в алгоритмі більше чотирьох цілих змінних не можна.

Протестувати правильну роботу цього алгоритму для цілих чисел: -2; -1; 0; 1; 4; 5; 6; 7.

Практична робота 4.

Лінійні алгоритмимовою Pascal

Завдання 1.(4 бали). Нехай R- це приблизна кількість риби у ставку(R>0), проте вона невідома. Одночасно виловити усю рибу в ставку неможливо. Тому першого дня зі ставка виловлюють K риб(K>0), помічають їх і відпускають назад у ставок. Через день знову закидають сітку і виловлюють M риб(M>0), серед яких виявляють N помічених риб(0<N<= K). Створіть і реалізуйте алгоритм, який знаходить приблизну кількість риб у ставку.

Розв'язання. Нехай у ставку R – риб, тоді K/R – це ймовірність виловити помічену рибу. Кількість помічених риб серед другого вилову приблизно дорівнює M*K/R =N. Звідси отримаємо приблизну кількість риб у ставку R =M*K/N риб.

programFisher; {назва алгоритму}

varr,k,m,n: integer; {оголошення змінних величин: **r,k,m,n** - це цілі числа}

begin {початок виконання дій алгоритму}

readln(k); {оголошення про введення числа **k** - це ціле число}

readln(m); {оголошення про введення числа **m** - це ціле число}

readln(n); {оголошення про введення числа **n** - це ціле число}

r:=k*mdivn; {арифметичні дії над цілими **k,m,n** і присвоєння результату **r**}

write(r); {оголошення про виведення числа **r** - це ціле число}

end. {кінець виконання дій алгоритму}

Протестуйте його для трійок цілих чисел (K;M;N)={(2000;2400;1000), (2300;2500;1600), (5581;3159;2992), (4581;3159;2992), (2001;20001;2001), (2077;2166;1552)}.

Завдання 2.(4 бали).Із молока, жирність якого становить a%(1<a<7) виготовляють сир жирністю b%(15<b<30).При цьому залишається сироватка жирністю c% (0,0001<c< 0,09).

Створіть і реалізуйте алгоритм, який знаходитькількість сиру **m кг**, що виходить із **k тонн** молока.

Розв'язання. Нехай із **k** тонн молока виходить m кг сиру. Маса жиру в k тоннах молока k *1000*(a/100) =10ka кг. Маса жиру в **m**кгсиру становитьm*(b/100) кг. Маса жиру в сироватц і становить(k*1000-m)*(c/100). Оскільки при переробці молока кінцевими продуктами є сир та сироватка, тоді складаємо рівняння для кількості жиру в обох продуктах: m*(b/100) + (k *1000-m)*(c/100) =10ka, звідси **m =1000k(a-c)/(b-c)**

programCheese; {назва алгоритму}

vara,b,c, k,m: real; {оголошення змінних величин: a,b,c, k,m: - це дійсні числа}

begin {початок виконання дій алгоритму}

writeln('введіть жирність молока 1<a<5 a='); readln(a);

writeln('введіть жирність сиру 15<b<30 b='); readln(b);

writeln('введіть жирність сироватки 0.001<c<0.100, c='); readln(c);

writeln('введіть кількість молока 1<k<1000, k='); readln(k);

{оголошення про введення числа k - це дійсне число}

m:=1000*k*(a-c)/(b-c); {арифметичні дії над дійсними a,b,c, k, і присвоєння m}

write(m); {оголошення про виведення числа m - це ціле число}

end. {кінець виконання дій алгоритму}.

Протестуйте алгоритм для четвірок дійсних чисел (a; b; c; k)={(5.5; 17.4; 0.1; 1), (3.89; 16.67; 0.086; 20), (4.581; 17.759; 0.029; 40.5), (3.181; 18.59; 39.92; 30), (4.1;19.61;0.08; 25).

Завдання 3.(4 бали).Із молока, жирність якого становить a%(1<a<7) виготовляють вершки **k кг** (1< **k** <100) вершків, жирністю b%(20<b<60). Самостійно створіть і реалізуйте алгоритм, який знаходить кількість молока **m кг**, жирність якого становить a%(1<a<7), із якого вийшло **k кг** (0.1< **k** <100) вершків, жирністю b% (20<b<60).

Розв’язання. Нехай із **m** кг молока виходить **k** кг вершків. Маса жиру в m кг молока **становить 0.01*a*m** Маса жиру в k**кг** вершків **становить 0.01*b*k** кг. Оскільки при переробці молока кінцевими продуктами є вершки, тоді складаємо рівняння для кількості жиру в обох продуктах: 0.01*a*m = 0.01*b*k. **Звідси m =b*k/a, для b,k,a дійсних.**

Протестуйте алгоритм для трійок дійсних чисел (a; b; k)={(5.5; 27.4; 1), (3.89; 26.67; 3.86), (4.581; 37.759; 20.5), (3.181; 28.59; 30), (4.1; 39.61;25).

Практична робота 5.

Лінійні алгоритмимовою Pascal

Завдання 1.(4 бали). Створіть і реаліз уйте алгоритм, який знаходить скільки треба досипати **a** кг солі до **k** кг водного **m%** розчину, щоб отримати водний розчин з концентрацією **n** %.

Розв’язання. Нехай у розчин треба досипати **a**кг солі. Маса солі у початковому розчині становить 0.01**mk** кг. Маса солі у новому розчині становить 0.01**n(k** +a). Тоді маємо рівняння: 0.01n(k +a)= 0.01mk+a. Звідси, маємо **a =0.01k(m-n)/(0.01n-1).**

programSaltSolution; { оголошення назви алгоритму}

vara,k,m,n: real; {оголошення змінних величин: a,k,m,n - це дійсні числа}

begin {початок виконання дій алгоритму}

writeln('введіть початкову масу води 1<k<5000 k='); readln(k);

writeln('введіть початковий відсоток концентрації солі у воді 1<m<10 m='); readln(m);

writeln('введіть кінцевий відсоток концентрації солі у воді 10<m<60 n='); readln(n);

a:=0.01*k*(m- n)/(0.01*n -1); {арифм-чні дії над дійсними k,m,n і присв-ня резуль-ту a}

write(a, 'кг'); {оголошення про виведення числа a - це ціле число}

end. {кінець виконання дій алгоритму}

Протестуйте його для трійок дійсних чисел (k;m;n)={(200;2;12), (300;2.5;16), (5500;3;29), (581;3;9), (200;1.8;20), (1000;1.6; 2)}.

Завдання 2.(4 бали).Швейна фабрика має пошити **k** костюмів двох моделей. Для визначення того, скільки костюмів і якої моделі треба пошити провели опитування серед покупців. **Результати опитування: 1-у модель вибрало m** покупців; **2-у модель вибрало n** покупців. Створіть і реалізуйте алгоритм, який знаходить кількість костюмів і якої моделі треба пошити, якщо опитано **m+n**покупців.

Розв'язання. Частка покупців, котрі вибрали першу модель, становить **m /(m+n).** Частка покупців, котрі вибрали другу модель, становить **n /(m+n).** Швейна фабрика має пошити:

1-у модель **km /(m+n) одиниць та** 2-у модель **kn /(m+n) одиниць.**

programModels; {оголошення назви алгоритму}

vara1,a2,k,m,n: integer; {оголошення змінних величин: **a1,a2**,n, k,m: - це цілі числа}

begin {оголошення початку виконання дій алгоритму}

writeln('введіть кількість замовлених костюмів 1<k<10000 a='); readln(k);

writeln('введіть кількість покупців 1-ої моделі 1<m<3000); readln(m);

writeln('введіть кількість покупців 2-ої моделі 1<n<3000); readln(n);

{оголошення про введення числа k - це дійсне число}

a1:=(k*m) div (m+n); {арифметичні дії над цілимин, k,m, і присвоєння результату **a1**}

a2:=(k*n) div (m+n); {арифметичні дії над цілимин, k,m, і присвоєння результату **a2**}

write('a1=',a1, 'одиниць; ', 'a2=', a2, 'одиниць.');{виведення кіль-сті костюмів}

end. {кінець виконання дій алгоритму}.

Протестуйте алгоритм для трійок цілих чисел (a; b; c; k)={(5; 1; 1), (3; 16; 20), (4581; 17;

29), (3181; 181; 59), (3900; 92; 30), (410;19;25).

Завдання 3.(4 бали).Син з батьком домовилися зустрітися між ***k*** та ***m*** годинамипротягом доби. Але у них існувала умова зустрічі: той, хто приходить першим на місце зустрічі, чекає другого не більше n хв, після чого покидає місце зустрічі. Самостійно створіть і реалізуйте алгоритм, який знаходить ймовірність зустрічі сина та батька.

Розв'язання. Нехай x – момент приходу сина; y – момент приходу батька; тоді умова зустрічі батька та сина записується виразом: |x-y|=<n. Тоді -n=< x-y=<n, звідки отримаємо: y=<x+ n; y>=x-n. Якщо A – подія, коли батько і син зустрінуться за цієї умови, то фактом зустрічі буде точка вибрана із заштрихованої 6-кутника. Тоді ймовірність зустрічі

P(A)=(Площа 6-кутника):(площа квадрата)=$(m-k)^2*60^2-(60m-60k-n)^2/((m-k)^2*60^2)$

program Meeting;

var k,m,n: integer; p: real;

begin writeln(' введіть нижню межу на проміжку зустрічі: 10<k<20 k='); readln(k);

writeln(' введіть верхню межу на проміжку зустрічі: 11<m<24 m='); readln(m);

writeln(' тривалість очікування в хвилинах 5<n<60 n='); readln(n);

p:=((m-k)*(m-k)*60*60-(60*m-60*k-n)*(60*m-60*k-n))/((m-k)*(m-k)*60*60);

write('p=',p); {виведення результату} **end.** {кінець алгоритму}

Протестуйте алгоритм для трійок дійсних чисел (k; m; n)={(15; 17.41), (8; 12; 80), (14; 17; 15), (18; 19;15), (12;15;25).

Практична робота 6.

Алгоритми геометричного змісту

Завдання 1. Створити, реалізувати алгоритм мовою Pascal, який знаходить кількість сторін опуклого багатокутника, якщо у багатокутника існує рівно k внутрішніх кутів, що дорівнюють ***a*** градусів(0<a<100), а усі решта внутрішніх кутів дорівнюють ***b***градусів(100<b <180).

program Poligon1;

var a, b, k, n: integer;

begin

a:=random(60); b:=100+random(20); k:= 1+random(3);

writeln('Кількість кутів, щодорівнюють k=', k);

writeln('Градусна міра декількох кутів, що дорівнюють A=', a);

writeln('Градусна міра декількох кутів, що дорівнюють B=', b);

n:=(360-k*(180-a)) div (180-b)+k;

writeln('Кількість сторін багатокутника n=', n); end.

Протестуйте правильність алгоритму: 1)a:=75; b:= 160; k:= 3; Відповідь: 5. 2) a:=80; b:= 160; k:= 3; Відповідь: 6. 3) a:=70; b:= 160; k:= 3; Відповідь: 4.

4)a:=90; b:=160; k:= 3; Відповідь: 7.

Завдання 2. Створити, реалізувати алгоритм мовою Pascal, який знаходить периметр та площу прямокутного трикутника, якщо у трикутника відомі довжини двох катетів.

program TRYkut2;

var a, b, p, s: real;

begin

a:=10+random(60); b:=10+random(40); writeln('Довжинапершогокатета, a=', a);

writeln('Довжинадругогокатета, b=', b); s:= a*b/2; p:=a+b+sqrt(a*a+b*b) ;

writeln('Периметр трикутника p=', p); writeln('Площа трикутника s=', s); end.

Протестуйте правильність алгоритму: 1)a:=75; b:=100; 2) a:=80; b:= 120; 3) a:=70; b:= 100; 4)a:=90; b:=180.

Завдання 3. Створити, реалізувати алгоритм мовою Pascal, який знаходить периметр та площу довільного трикутника, якщо відомо у трикутники довжини трьох сторін.

program TRYkut3;

var a, b,c, p, s: real;

begin

a:=10+random(60); b:=10+random(40); c:=10+random(50);

writeln('Довжина першої сторони, a=', a); writeln('Довжина 3-ої сторони, c=', c);

writeln('Довжина другої сторони, b=', b); p:= (c+a+b)/2;

s:=sqrt(p*(p-a)*(p-b)*(p-c)) ; p:= 2*p;

writeln('Периметр трикутника p=', p); writeln('Площа трикутника s=', s); end.

Протестуйте правильність алгоритму: 1)a:=75; b:=100; c=85; 2) a:=80; b:= 120; c =95; 3) a:=70; b:= 100; c=105; 4)a:=190; b:=180, c=185;

Завдання 4. Є коло радіуса R з координатами центра (x,y) і пряма, що задана координатами двох своїх точок. Якої довжини відрізок прямої лежить всередині кола?

Вхідні дані. Задано рядок з 7-ми чисел: радіус кола, координати (x, y) центра і координати 2-х точок прямої. Всі числа цілі, за абсолютним значенням не перевищують 10000.

Вихідні дані.Вивести шкану довжину з точністю до 5 цифр після коми. Якщо коло і пряма не перетинаються, вивести -1, якщо дотикаються - вивести 0.

Розв'язання

Взаємне розташування кола і прямої має декілька випадків. Знаходимо відстаньL від центру кола до прямої, яка задана двома точками. Якщо відстань більша за радіус, то пряма і коло не перетинаються, коли ж рівні, то дотикаються, а інакше – слід знайти довжину хорди, яка сполучає дві точки перетину прямої і кола. Враховуємо, що відстань обчислена наближено. Для знаходження точок перетину прямої і кола – використаємо формулу.

```
PROGRAm CIRCUS2;

var r,x0,y0,x1,y1,x2,y2: integer;

a,b,c,aa,bb,cc,x3,y3,x4,y4,L,d,d1: real;

begin

readln(r,x0,y0,x1,y1,x2,y2);

L:=abs((x2-x1)*(y0-y1)-(y2-y1)*(x0-x1) )/sqrt(sqr(x2-x1)+sqr(y2-y1));

if (L-r)>0.00001 then writeln(-1)  else  if abs(L-r)<=0.00000005 then writeln(0)

else  begin  a:=y2-y1;  b:=x1-x2;  c:=x1*(y1-y2)+y1*(x2-x1);

if abs(b)<=0.000000001 then  begin  y3:=-sqrt(r*r-sqr(c/a+x0))+y0;

 y4:=sqrt(r*r-sqr(c/a+x0))+y0;  x3:=-c/a;  x4:=-c/a;

d:=sqrt(sqr(x4-x3)+sqr(y4-y3));  writeln(d:0:5);  end else  begin  aa:=a*a/(b*b)+1;
bb:=(y0+c/b)*a/b-x0;  cc:=x0*x0+sqr(y0+c/b)-r*r;

d1:=bb*bb-aa*cc; x3:=(-bb-sqrt(d1))/aa  x4:=(-bb+sqrt(d1))/aa; y3:=(-a*x3-c)/b;

y4:=(-a*x4-c)/b;  d:=sqrt(sqr(x4-x3)+sqr(y4-y3));  writeln(d:0:5);  end;  end;  end.
```

Дані для тестування алгоритму:

А)Введення: 5 0 0 4 1 4 2 Виведення результату : 6.

Завдання 5.Створити та реалізувати алгоритм мовою Pascal, що визначає за відомими довжинами трьох сторін трикутника чи має він внутрішній прямий кут.

```
PROGRAm LINEAR1;

var a,b,c,k:integer;

begin  readln(a, b, c);

while (a<>0)and(b<>0)and(c<>0) do begin      if c<a then   begin k:=c; c:=a; a:=k;

if c<b then begin k:=c; c:=b;b:=k; end; end;   if c<b then begin k:=c; c:=b; b:=k;

if c<a then begin k:=c; c:=a; a:=k; end; end;   if (a*a+b*b=c*c) then writeln(' yes')

else  if (a*a+b*b<>c*c) then writeln('nou');   writeln('    ', a, '    ', b, '    ', c);  end;  end.
```

Дані для тестування алгоритму:

А)Введення: 5 4 3 Виведення результату : yes.

Б)Введення: 5 12 13 Виведення результату : yes.

В)Введення: 25 24 7 Виведення результату : yes.

Г)Введення: 5 4 6 Виведення результату : nou.

Завдання 6.Створити та реалізувати алгоритм мовою Pascal, що визначає за відомими довжинами **a**, **b**, **c**, **d** чи можна у прямокутник зі сторонами **a**, **b** цілком помістити прямокутник зі сторонами **c**, **d**.

```
Program fiercutnyk2;

const eps=0.000001;  var  a,b,c,d,hk,wk,hl,wl,ab,cd,bc,ad,af,ge,ga,ae,ac,cf,

bae, da, baf, bac, caf, cab, fac: real;   s:string;

begin   readln(a, b, c, d);

if a<b then  begin  wk:=b;  hk:=a;  end   else  begin  wk:=a;  hk:=b;  end;

if c<d then  begin  wl:=d;  hl:=c;  end   else   begin   wl:=c;   hl:=d;  end;

if hl>hk then s:='NO'   else   begin  if wk>=wl then s:='YES'  else  begin

ab:=wl; cd:=wl;  bc:=hl; ad:=hl;  af:=hk;   ac:=sqrt(wl*wl+hl*hl);

cf:=sqrt(ac*ac-hk*hk);  cab:=arctan(bc/ab);  fac:=arctan(cf/af);

bae:=3.1415926/2-cab-fac;  ge:=ad*sin(bae)+ab*cos(bae);

if (wk-ge)>=eps then s:='YES' else s:='NO';    end;  end;   writeln(s); end.
```

Дані для тестування алгоритму:

А)Введення: 5 4 1 1 Виведення результату : yes.

Б)Введення: 5 8 1 3 Виведення результату : yes.

В)Введення: 25 24 3 5 Виведення результату : yes.

Г)Введення: 5 4 6 8 Виведення результату : nou.

Завдання 7. Створити та реалізувати алгоритм мовою Pascal, що визначає координати вершини Anправильного трикутника AnBnCn, якщо $A_1(0;1)$; $B_1(-0,5; 0)$; $C_1(0,5; 0)$ і кожна сторона наступного рівностороннього трикутника збільшена на 2 одиниці довжини. Послідовність правильних трикутників показана на малюнку.

```
ProgramTryangul;
var an, a1, d, n, x, y: real; begin readln(n); a1:=1; d:=2; an:=a1+d*(n-1);
x:=(a1+an)*n/2-0.5-an/2; y:=sqrt(3)*an/2; writeln(x:0:3,' ',y:0:3); end.
```

Дані для тестування алгоритму:

N= 20; N= 25; N= 50; n=76.

Завдання 4. Самостійно створити, реалізувати алгоритм мовою Pascal, який знаходить периметр та площу довільного прямокутника, якщо відомо у нього довжини двох сусідніх сторін.

Практична робота 7.

Лінійні алгоритмимовою Pascal

Завдання 1. На конференцію прибуде **k** делегацій, у кожній з них ***m*** осіб. Організаторам конференції треба **скласти і реалізувати алгоритм, який** знаходить кількість усіх зустрічей, які можуть відбутися між: а) двома делегаціями конференції; б) між двома особами, які є учасниками конференції; в) між трьома делегеціями; г) між трьома особами.

```
program Conferenz1;                    {оголошення назви алгоритму}
var k,m,n: integer;                    {оголошення змінних цілих величин k,m,n для алгоритму}
```

begin writeln(' введіть кількість делегацій: 4<k<10 k='); readln(k); {ведення числа k }

writeln(' введіть кількість осіб в делегації: 3<m<12 m='); readln(m); {введення m}

n:=(k-1)*kdiv 2; **{**обчислення кількості усіх двосторонніх зустрічей**}**

writeln('кількість усіх двосторонніх зустрічей між'**, k,** 'делегаціями: n='**, n);**

n:=(m*k-1)*m*kdiv 2; **{**обчислення кількості усіх двосторонніх зустрічей між особами **}**

writeln('кількість усіх двосторонніх зустрічей між'**, m*k,** 'учасниками: n='**, n);**

n:=(k-2)* (k-1)*kdiv 6; **{**обчислення кількості усіх тристоронніх зустрічей **}**

writeln('кількість усіх тристоронніх зустрічей між'**, k,** 'делегаціями: n='**, n);**

n:=(m*k-2)* (m*k-1)*m*kdiv 6; **{**обчислення кількості усіх тристоронніх зустрічей **}**

writeln('кількість усіх тристоронніх зустрічей між'**, m*k,** 'учасниками: n='**, n); writeln; end.**

Протестуйте алгоритм для пар чисел (k; m)={(15; 17), (8; 12), (14; 15), (18; 10), (12;15;25).

Завдання 2. На конференцію прибуде **k** делегацій. Організаторам конференції треба **скласти і реалізувати алгоритм, який** знаходить тривалість в годинах конференції, якщо на конференції відбудуться лише тристоронні зустрічі між делегаціями, і на кожну таку зустріч разом з перервою витрачається ***m*** хвилин, протягом цих хвилин одночасно можуть відбутися тільки ***n*** зустрічей. Врахуовуйте, що по ***p*** хвилин проходитиме відкриття та закриття конференції.

program Conferenz2; {оголошення назви алгоритму}

var k,m,n,p: integer; {оголошення змінних цілих величин k,m,n для алгоритму}

begin writeln(' Введіть кількість делегацій: 4<k<20 k='); readln(k); {введення k}

writeln(' Введіть тривалість однієї зустрічі: 4<m<60 m='); readln(m); {введення m}

writeln(' Введіть кількість одночасних зустрічей: 3<n<20 n='); readln(n); { n}

writeln(' Введіть тривалість відкриття і закриття 15<p<60 p='); readln(p); { p}

n:=mdiv 30 +n*(k-2)* (k-1)*kdiv (360*p); **{**обчислення часу конференції **}**

writeln('Тривалість конференції n='**, n, 'год');**

writeln; end.

Протестуйте алгоритм для четвірок чисел (k; m; n; p)={(9; 20; 3; 30)=10 год, (10; 15; 3; 20) =10 год, (20; 20; 20; 30)=20 год,(15; 30; 5; 30)=10 год}.

Завдання 3. На бізнес-конференцію прибуде **k** делегацій, із яких тільки **m делегацій матимуть лише тристоронні зустрічі, а усі інші учасники матимуть тільки двосторонні зустрічі.** Організаторам конференції треба **скласти і реалізувати алгоритм, який** знаходить суму усіх внесків на банківський рахунок конференції, якщо на конференції усі делегеції за усі свої тристоронні зустрічі між делегаціями одноразово сплачують **n**доларів, і усі делегеції за усі двосторонні зустрічі одноразово сплачують **p** доларів, Враховуйте, що на конференції не існує делегацій, у яких не проходитиме зустрічей на конференції.

program Conferenz3; {оголошення назви алгоритму}

var k,m,n,p: integer; {оголошення змінних цілих величин k,m,n }

begin writeln(' Введіть кількість делегацій: 4<k<20 k='); readln(k); {введення k}

writeln(' Введіть кількість делегацій для тристорон. зустрічей: 4<m<20 m='); readln(m);

writeln(' Введіть тариф за тристоронні зустрічі: 1000<n<10000 n='); readln(n);

writeln(' Введіть тариф за двосторонні зустрічі: 1500<p<60000 p='); readln(p);

n:=m*n + (k-m)*p; {обчислення суми внесків конференції **}**

writeln('суми внесків від делегацій конференції n=**', n, 'доларів');**

writeln; end.

Протестуйте алгоритм для четвірок чисел (k; m; n; p)={(20; 5;1000; 30000)=455 000 дол, (7; 3; 2000; 5000)=20 000 дол, (15; 9; 1200; 3700)=33000 дол, (18; 10; 2500; 3000)=49000 дол}.

Завдання 4. (3 бали).Самостійно скласти і реалізувати алгоритм, що знаходить окремо кількість чотиристоронніх договорів(кожна делегація за результатами тристоронньої зустрічі отримує два лише договори) і окремо кількість двосторонніх договорів, які отримують учасники конференції при умові завдання 3.

Практична робота 8.

Лінійні алгоритми алгебраїчного змісту

Завдання 1. Створити, реалізувати алгоритм мовою Pascal, який знаходить значення функції $f(x)=a*x^8 + b*x^6 - c*|x^3-x^5|$, якщо x, a, b, c – випадкові цілі числа.

program Polinom1;

var a, b, c, x, f: integer;

begin a:=-random(60); b:=100-random(20); c:= 10-random(300);x:= 10-random(20);

writeln('a=', a); writeln('b=', b); writeln(' c=', c); writeln(' x=', x);

f:=a* x* x* x* x* x* x+b* x* x* x* x* x* x –c*abs(x* x* x- x* x* x* x* x);

writeln('Значення функції для випадкових чисел f=', f); writeln('**'); end.**

Протестуйте правильність алгоритму: 1)a:=75; b:= 160; c:= 2;x =7.

2) a:=80; b:= 160; c:= 3; x=-1. 3) a:=70; b:= 160;c:= -7; x= 4.

4)a:=90; b:=160; c:= 9; x=17.

Завдання 2. Створити, реалізувати алгоритм мовою Pascal, який знаходить значення функції f(x)= якщо x, a, b, c – випадкові додатні дійсні числа.

program Drob2;

var a, b, c, x, f: real;

begin

a:=1+random(60); b:=12+random(30); c:= 11+random(40);x:= 10+random(20);

writeln('Число a=', a); writeln('Число b=', b); writeln('Число c=', c);

writeln('Число x=', x);

f:=sqrt((a+c)*x*x*x+b*x*x+x)/((c+b)*x*x*x+a*x*x+x);

writeln('Значення функції для випадкових чисел f=', f); writeln('**'); end.**

Протестуйте правильність алгоритму:1)a:= 5; b:= 6; c:= 2;x =7. 2) a:=8; b:= 1; c:= 3; x=1. 3) a:=7; b:= 5;c:= 7; x= 4. 4)a:=9; b:=3; c:= 8; x=2.

Завдання 3. Створити, реалізувати алгоритм мовою Pascal, який знаходить значення функції f(x)= якщо x, a, b, c – випадкові від'ємні дійсні числа.

program Drob3;

var a, b, c, x, f: real;

begin

a:=-1-random(60); b:=-12-random(30); c:=- 11-random(40);x:= -10-random(20);

writeln('Число a=', a); writeln('Число b=', b); writeln('Число c=', c);

writeln('Число x=', x);

f:=(a/sqrt(x*x+2))-(b/ sqrt(x*x+3)) +(c/ sqrt(x*x+4));

writeln(‘Значення функції для випадкових чисел f=’, f); writeln(‘**’); end.**

Протестуйте правильність алгоритму: 1)a:=-75; b:=-100; c=-85; x =-7. 2) a:=-80; b:= -120; c=-95; x =-37. 3) a:=-70; b:= -100; c=-105; x =-27. 4)a:=-190; b:=-180, c=-185; x =-17.

Завдання 4. Самостійно створити, реалізувати алгоритм мовою Pascal, який знаходить периметр та площу довільного прямокутника, якщо відомо у нього довжини двох сусідніх сторін.

Практична робота 9.

Лінійні алгоритми мовою Pascal

–

Завдання 1. На конференцію приїхали рівно по k представниківвідm(не менше ніж 2) фірм-конкурентів по виробництву гри "Campf", при цьому, усі представники різних фірм є конкурентами. Відомо, що у кожного учасника конференції рівно nконкурентів серед усіх інших учасників. Скласти алгоритм, який знаходить найбільшу кількість учасників та найменшу кількість фірм-конкурентів в конференції?

Розв’язання. Кількість учасників конференції дорівнює km осіб. З іншого боку, кількість конкурентів у одного представника дорівнює k(m-1)= n, звідси маємо рівність km = n + k, права частина якого є лінійний вираз відносно двох змінних. Лінійний вираз p(k)= n + k досягає свого найбільшого і найменшого значення на межах числового проміжка від 1 до n. Якщо k = n, то маємо найбільше значення p(n)= n + n =2n, тому nm = 2n, звідси m = 2. Відповідь: 2n осіб, 2 фірми.

Алгоритм мовою Pascal

program MinConkurent; vark,m,n:integer; begin

writeln('введіть число конкурентів в особи 2<n<10^9 n='); readln(n);

write('найбільше:',2*n, ' осіб'); {вивід найбільшого числа учасників}

write('найменше: 2 фірми'); {вивід найменшого числа учасників}

end. {оголошення кінця алгоритму}

Завдання 2. На конференцію приїхали рівно по k представниківвідm(не менше ніж 2) фірм-конкурентів по виробництву гри "Campf", при цьому, усі представники різних фірм є конкурентами. Відомо, що у кожного учасника конференції рівно nконкурентів серед усіх інших учасників. Скласти алгоритм, який знаходить найбільшу кількість фірм-конкурентів і найменшу кількість представників в конференції?

Розв’язання. Кількість учасників конференції дорівнює km осіб. З іншого боку, кількість конкурентів у одного представника дорівнює k(m-1)= n, звідси маємо рівність km = n + k, права частина якого є лінійний вираз відносно двох змінних. Лінійний вираз p(k)= n + k

досягає свого найбільшого і найменшого значення на межах числового проміжка від 1 до n. Якщо у формулі p(k)= n + k, підставимо k = 1, то отримаємо найменше значення p(1)= n + 1 , тому найбільше значення m = n+1, звідси n = m-1. Відповідь: n+1 фірм, 1 особа.

Алгоритм мовою Pascal

program MaxConkurent; varn:integer; begin

writeln('введіть число конкурентів в особи 2<n<10^9 n='); readln(n);

write('найбільше:' n+1, 'фірм'); {вивід найбільшого числа фірм}

write('найменше: 1 представник від фірми'); {вивід найменшого числа представників } **end.** {оголошення кінця алгоритму}

Завдання 3. Відомо, що книжкова полиця вміщає kоднакових товстих книг, але k+1-а книга вже не влазить. Так само на неї можна поставити mоднакових тонких книг, am+1 -а вже не влізе. Скласти алгоритм, який знаходить можливість, щоб на полиці помістилися одночасно: nтовстих тар тонких книг.

Розв'язання. Якщо числовий вираз n/k + p/m <=1, то можна, якщо числовий n/k + p/m > 1, то не можна помістити одночасно книги на полицю.

Алгоритм мовою Pascal (використовує повне розгалуження, «якщо-то, інакше»)

program BIBLIO; var k,m,n,p,h:real; begin

writeln('введіть число товстих книг 2<k<10^9 k='); readln(k);

writeln('введіть число тонких книг 2<m<10^9 m='); readln(m);

writeln('введіть даних товстих книг 2<n<10^9 n='); readln(n);

writeln('введіть даних тонких книг 2<p<10^9 p='); readln(p); h:=(n/k)+(p/m);

if (h<1) or (h=1) then**write(' кнгиги можна помістити')** {розгалуження для результату}

else w**rite('не можна помістити книги');** {інакше то вивід результату не можна}**writeln ('h=', h); end.**

Завдання 4. Самостійно скласти і реалізувати алгоритм впорядкування виразів: **(n/k)+(p/m)-(m/p)-(k/m); та (p/k)+(n/m)-(k/n)-(n/p) в порядку зростання для дробових чисел k,m, n,p.**

Завдання 5. Самостійно скласти і реалізувати алгоритм впорядкування виразів:**(1/k)+(1/m)-1/p)-(1/n); та (1/k)+(1/p)-(1/m)-(1/n) в порядку зростання для дробових чисел k,m,n, p.**

Практична робота 10.

Лінійні алгоритми мовою Pascal

Завдання 1. Створити та реалізувати мовою програмування Pascalвипадкове трицифрове число і розділити його на цифри. Оскільки ми користуємося позиційною десятковою системою, то поділ числа на цифри виконується цілочисельним діленням даного числа на 10. Для цього використовуємо операції DIV та MOD.

ProgramPROBA_01;

vara:integer;

begin a:=1+random(999);

writeln(a div 100, ' ' , (a mod 100) div 10,' ',a mod 10); end.

Завдання 2. Створити та реалізувати мовою програмування Pascal, що знаходить найменшу кількість розрізів куба розміром AxBxC, де 1 ≤ A, B, C ≤ 200000. на одиничні кубики розміром 1x1x1. Найменша кількість розрізів не залежить від вибору порядку прорізання сторін і обчислюється за формулою. Виведемо її. Почнемо розрізати по стороні A і одразу проріжемо на максимальну кількість частин. Отримаємо **A** частин розміру **1*B*C** і виконаємо **A-1** розрізів. Далі ріжемо кожен з отриманих кусків по стороні B. Отримаємо **A*B** кусків розмірами **1*1*C**, виконавши при цьому всього **A*(B-1)** розріз. Залишається виконати розрізання по стороні C. Буде виконано **A*B*(C-1)** розрізів. Усього маємо **(A-1)+A*(B-1)+A*B*(C-1)=A-1+A*B-A+A*B*C-A*B=A*B*C-1** розрізів.

Program PROBA_02;

var a,b,c: integer; begin a:=abs(random(9999));

b:=abs(random(999999)); c:=abs(random(9999));

writeln(abs((a*b*c)-1)); end.

Завдання 3. Створити та реалізувати мовою програмування Pascal, що знаходить найбільшу кількість точок із цілочисельними координатами на аркуші в клітинку можна накрити квадратом зі стороною **N** клітинок(**1 ≤ N ≤ 10000**). Накреслимо на папері в клітинку квадратики зі сторонами 1, 2, 3, ... клітинок. У першому випадку ми закриємо 4 точки, у другому – 9, у третьому – 16... Як бачимо, ми отримуємо числа, що є точними квадратами: 2^2, 3^2, 4^2... Тому для введеного числа A потрібно вивести квадрат наступного числа.

Program PROBA_03;

var a: integer;

begin a:=abs(random(999999)); writeln((a+1)*(a+1)); end.

Завдання 4. **N** будинків вишикувані в ряд.Створити та реалізувати мовою програмування Pascal, що знаходить кількість способів поселити Віку та Юлю в різні будинки так, щоб ці будинки не були сусідніми.

Program PROBA_04;

```
var a: integer;
begin a:=abs(random(999999));
writeln((a-2)*(a-1));  end.
```

Зміст

Зміст

1.Лінійні алгоритми мовою Pascal

Практична робота 1. Лінійні алгоритми мовою Pascal.

Практична робота 2. Лінійні алгоритми мовою Pascal.

Практична робота 3. Лінійні алгоритми мовоюPascal.

Практична робота 4. Лінійні алгоритми мовою Pascal.

Практична робота 5. Лінійні алгоритми мовою Pascal.

Практична робота 6.Алгоритми геометричного змісту.

Практична робота 7. Лінійні алгоритмимовою Pascal.

Практична робота 8. Лінійні алгоритми алгебраїчного змісту.

Практична робота 9. Лінійні алгоритми мовою Pascal.