МІНІСТЕРСТВО АГРАРНОЇ ПОЛІТИКИ ТА ПРОДОВОЛЬСТВА УКРАЇНИ
МИКОЛАЇВСЬКИЙ НАЦІОНАЛЬНИЙ АГРАРНИЙ УНІВЕРСИТЕТ
Навчально-науковий інститут економіки та управління

ІНФОРМАЦІЙНІ СИСТЕМИ ТА ТЕХНОЛОГІЇ

МЕТОДИЧНІ РЕКОМЕНДАЦІЇ
для виконання лабораторних робіт
студентами денної та заочної форм навчання
напряму підготовки 6.030601 «Менеджмент»

Розділ «***Мова програмування Сі***»

Миколаїв
2014

УДК 004.43
ББК 32.973-01
І-95

Друкується за рішенням науково-методичної комісії обліково-фінансового факультету Миколаївського національного аграрного університету, протокол № 10 від 17.06.2014 року.

Укладачі:

Ш. М. Іхсанов – кандидат технічних наук, старший науковий співробітник, доцент, зав. кафедри інформаційних систем і технологій Миколаївського національного аграрного університету;

В. В. Лопушанська – кандидат економічних наук, старший викладач кафедри інформаційних систем і технологій Миколаївського національного аграрного університету.

Рецензенти:

М. Т. Фісун – доктор технічних наук, професор, завідувач кафедри інтелектуальних інформаційних систем Чорноморського державного університету імені Петра Могили;

В. І. Гавриш – доктор економічних наук, професор кафедри тракторів та сільськогосподарських машин інженерно-енергетичного факультету Миколаївського національного аграрного університету.

ВСТУП 5
1. ПОНЯТТЯ ЗМІННОЇ 5
2. АРИФМЕТИЧНІ ОПЕРАЦІЇ 6
3. ЛОГІЧНІ ОПЕРАЦІЇ 7
4. ОСНОВНІ ОПЕРАТОРИ (КЛЮЧОВІ СЛОВА) 8
4.1. Арифметичні оператори 8
4.2. Оператор умовного переходу if 8
4.3. Оператор циклу for 9
4.4. Оператор безумовного переходу goto 10
4.5. Оператор циклу while 11
4.6. Ключові слова break і continue 12
5. ОСНОВНІ ПРАВИЛА СКЛАДАННЯ ПРОГРАМ 14
5.1. Підключення необхідних бібліотек мови Сі 14
5.2. Ключове слово main() 14
5.3. Опис змінних 14
5.4. Реалізація основних математичних функцій мови Сі 15
6. МАСИВИ 18
6.1. Опис масивів 18
6.2. Конструкція #define для оголошення розмірності масивів 19
6.3. Ініціалізація масивів при оголошенні 20
7. ФУНКЦІЯ PRINTF() 21
7.1. Виведення на екран заданого тексту 21
7.2. Виведення на екран значень змінних 21
8. ПРИКЛАДИ ПРОГРАМ 23
8.1. Друк координат вектора, які перевищують заданий поріг 23
8.2. Множення двох прямокутних матриць 23
8.4. Розкладення числа на прості множники 24
8.5. Рішення фізичної задачі 26
9. РОБОТА У ІНТЕГРОВАНОМУ СЕРЕДОВИЩІ ПРОГРАМУВАННЯ C++Builder6 28
9.1. Запуск програми та вибір проекту 28
9.2. Компіляція і налагодження програми 31
9.3. Використання режиму налагодження (debugging) 32
10. ВВЕДЕННЯ ІНФОРМАЦІЇ ДО ПРОГРАМИ З ЕКРАНУ 34
10.1. Функція scanf() 34
10.2. Приклади програм, що використовують введення даних з екрану 35
10.2.1. Введення вектора з екрану. 35
10.2.2. Введення дати з екрану 36
10.2.3. Доопрацювання програми рішення фізичної задачі з розділу 8.5 38
11. СТВОРЕННЯ ФУНКЦІЙ КОРИСТУВАЧА 40
11.1. Загальні правила написання функцій користувача 40
11.2. Приклад використання функцій користувача 41
12. ВПРАВИ ДЛЯ САМОСТІЙНОЇ РОБОТИ 43
13. ЗАДАЧІ ДЛЯ САМОСТІЙНОГО СКЛАДАННЯ ПРОГРАМ 59
13.1. Задачі І-го рівня складності 59
13.2. Задачі середньої складності 64
13.3. Задачі підвищеної складності 72
14. РОЗПОДІЛ МАТЕРІАЛУ НА МОДУЛІ 83
СПИСОК ЛІТЕРАТУРИ 84

ВСТУП

"Сі - це досить виразна мова програмування, призначена для опису широкого кола задач, яка містить сучасні механізми керування обчислювальним процесом і роботи з даними. У той же час мова Сі дуже проста... Сі - проста, витончена мова програмування, на якій зупиняють свій вибір усе більше число програмістів" [1].

На даний час мова програмування Сі (C++) є найбільш поширеною у світі разом з мовами, які виросли із неї (C#, Objective-C, Java, JavaScript, PHP).

Запропонований курс мови Сі є досить обмеженою початковою частиною мови, однак її засвоєння дозволяє вирішувати широке коло задач. Засвоєння цього курсу, як показує досвід викладання, проходить не гірше, ніж засвоєння мови Бейсик, який широко викладають у середній школі. У той же час, при виникненні більш глибокого інтересу до програмування, перед студентами відкриваються широкі перспективи професійного освоєння програмування.

1. ПОНЯТТЯ ЗМІННОЇ

Визначення. Змінна – це область у оперативній пам'яті комп'ютера, що може змінюватися під час виконання програми.

Змінна виникає в момент присвоєння їй імені автором програми.

Визначення. Ім'я змінної – це будь-який набір латинських літер, цифр і знака підкреслення, що починається з літери (великої або малої).

Приклади імен: **`a, b, x, y, z, size1, matrix, a123, COPY_PUT`**. На відміну від математики, де імена об'єктів складаються, як правило, з однієї літери, у програмуванні імена змінних і інших об'єктів можуть бути довгими і, як правило, несуть інформацію про їхнє призначення.

Зауваження. Мова Сі розрізняє регістри, використані в імені змінної, тобто змінна **`x`** ***і*** **`X`**, **`matrix`** ***і*** **`Matrix`** - ***це різні змінні!***

У мові Сі введено наступні основні типи змінних:

- тип **char** (скорочення від англійського слова character - символ) - **символьна** змінна, яка займає **один байт**, діапазон зміни **[-128, +127]**, якщо знак числа не потрібен, то використовується тип **unsigned char** (unsigned – без знаку) с діапазоном зміни **[0, 255]**;
- тип **int** (скорочення від **integer** - цілий) - ціла змінна, займає **4 байта**, діапазон зміни **[-2147483648, +2147483647]**, відповідно, **unsigned int** змінюється у діапазоні **[0, 4294967295]**;
- тип **float** (**float** - плаваючий) - змінна з плаваючою комою, займає 4 байти, представлена в експонентній формі $m*2^p$, де p - порядок зі знаком що займає 1 байт і m - мантиса зі знаком, що займає 3 байти, діапазон зміни $\left[1.18\cdot10^{-38},3.40\cdot10^{-38}\right]$;
- тип **double**(double - подвійний) - змінна з плаваючою комою з подвійною точністю, займає **8 байт**, з них **2 байти** займає порядок зі знаком та **6 байт** мантиса зі знаком, діапазон зміни $\left[2.23\cdot10^{-308},1.78\cdot10^{308}\right]$;

2. АРИФМЕТИЧНІ ОПЕРАЦІЇ

= - присвоєння,

+ , − - додавання, віднімання,

***** - множення,

/ - ділення,

% - залишок від ділення (модуль),

++ - збільшення на одиницю (інкремент),

−− - зменшення на одиницю (декремент).

Зауваження 1. На відміну від математики, знак множення у всіх мовах програмування не може бути опущено. Як і у математиці, для зміни порядку обчислень використовують дужки, але на відміну від математики, де для цієї мети можуть використовуватися квадратні і фігурні дужки, у мовах програмування можна використовувати тільки круглі дужки.

Зауваження 2. Цілу частину числа від дробової у мові Сі відокремлюють крапкою!

Приклади арифметичних виразів:

$\frac{b^2-4ac}{2a}$ ~ **(b*b-4*a*c)/(2*a),**

$\frac{\frac{1}{a}-\frac{1}{b+c}}{\frac{1}{a}+\frac{1}{b+c}}$ ~**((1/a-1/(b+c))/((1/a+1/(b+c))**,

$\frac{4}{a+\frac{1}{b+\frac{1}{c}}}:\frac{1}{a+\frac{1}{b}}-\frac{4}{b(abc+a+c)}$

~ **(4/(a+1/(b+1/c))/(1/(a+1/b))-4/b/(a*b*c+a+c)**.

3. ЛОГІЧНІ ОПЕРАЦІЇ

== - порівняння на рівність,

!= - порівняння на нерівність,

> - більше,

>=- більше або дорівнює,

<- менше,

<= - менше або дорівнює,

&& - логічне "і",

||- логічне "або", (**&** - знак амперсанда).

Приклади логічних виразів:

$x \le 5$ ~ **x <= 5,** $-3 \le x \le 5$ ~ **x >= -3 && x <= 5,**

$x \ne 5$ ~ **x != 5,** $x \le -12 \cup x \ge 7$ ~ **x <= -12 || x >= 7.**

4. ОСНОВНІ ОПЕРАТОРИ (КЛЮЧОВІ СЛОВА)

4.1. Арифметичні оператори

Визначення. Арифметичним оператором є операція присвоєння змінній заданого арифметичного виразу.

Зауваження. Мовою Сі наприкінці оператора завжди ставлять крапку з комою (;) .

Приклади:

```
d=(b*b-4*a*c)/(2*a); x1=(-b+d)%2; a+=b; a-=b; a*=b;
a/=b; i++; i--;.
```

Оператори: **`i++; i--; a+=b; a-=b; a*=b; a/=b;`** — коротка форма для операторів: **`i=i+1, i=i-1; a=a+b; a=a-b; a=a*b; a=a/b;`**.

4.2. Оператор умовного переходу if

Синтаксис оператору:

`if` (логічний вираз) оператор 1 **`else`** оператор 2 ,

або коротка форма:

`if` (логічний вираз) оператор 1.

Тут **`if`** і **`else`** ключові слова, у перекладі з англійського "якщо" і "інакше". Якщо логічне вираження, поміщене після ключового слова **`if`** у круглих дужках, є істиною, то виконується оператор 1, інакше виконується оператор 2, або, при використанні короткої форми, керування передається операторові, який знаходиться за оператором **`if`**.

Приклади:

```
if(a > 0) a++; else a=b;
if(b*b-4*a*c < 0) prizn=1;
```

4.3. Оператор циклу for

Синтаксис оператору:

for (оператор 1; умова; оператор 2)
    оператор 3 (тіло циклу);

У перекладі з англійського for означає "для". Оператор працює в такий спосіб:

Спочатку виконується оператор 1, потім перевіряється умова, і, якщо вона вірна, то виконується оператор 3, потім виконується оператор 2 і потім знову перевіряється умова, потім знову оператор 3, за ним оператор 2, і так повторюється циклічно доти, поки умову не буде виконано. Ця логіка виконання оператора **for** графічно відображена на рис. 4.1.

Рис. 4.1. Логіка виконання циклу **for**

За приклад розглянемо по кроках наступний оператор циклу, що реалізує обчислення факторіала від даного числа:

```
n=4; nf=1;
for(i=2; i <= n; i++)
  nf*=i;
```

1-ий крок: **i=2, 2 <= 4** - вірно, **nf=1*2=2, i=2+1=3;**

2-ий крок: **3 <= 4** - вірно, **nf=2*3=6, i=3+1=4;**

3-ій крок: **4 <= 4** - вірно, **nf=6*4=24, i=4+1=5;**

4-ий крок: **5 <= 4** - не вірно, кінець циклу.

Отже, наприкінці виконання цього оператора циклу ми маємо: **nf=n!(факторіал), i=n+1.**

4.4. Оператор безумовного переходу goto

Синтаксис оператору:

Мітка:

goto Мітка;

Міткою є будь-яка послідовність символів, що задовольняє вимогам, які висуваються до імені змінної, після якої поставлено двокрапку (:), в імені мітки рекомендується використовувати великі букви. Мітку не можна використовувати в якості змінної! У перекладі з англійського go to означає "йди до", у мові Сі пишеться разом.

Програма, зустрівши оператор **goto**, передає керування на рядок, напочатку якого стоїть зазначена мітка, мітка може стояти, як після оператора **goto**, так і до нього.

Нижче наводиться приклад реалізації за допомогою оператора **goto** фрагмента програми обчислення факторіалу, наведеної у розділі 4.3.

```
n=4; nf=1;
i=2;
M: if(i < n)
     { nf*=i;
       i++;
       goto M;
     }
```

Зауваження. У цій невеликій програмі використано одне з важливих угод мови Сі, які необхідно засвоїти. Відповідно до синтаксису операторів if і for, після ключового слова виконується усього лише один оператор, і не вказано, що робити, якщо потрібно виконати декілька операторів. Виявляється, якщо групу операторів узяти у фігурні дужки, то ця група стає одним (складеним) оператором,

іноді використовують термін «блок операторів» Кількість поєднуваних операторів ніяк не обмежується.

В якості вправи розпишіть по тактах виконання наведеного фрагмента.

4.5. Оператор циклу while

Синтаксис оператору:

1 форма: **`while`** (умова) оператор;

2 форма: **`do`** оператор **`while`** (умова);

У перекладі з англійської “while” означає "поки", do - "робити". Оператор **`while`** також можна записати через оператори **`if`** і **`goto`**:

1 форма:

```
M: if(условие)
     { оператор;
       goto M;
     }
```

2 форма:

```
M: оператор;
   if(умова)
     goto M;
```

Таким чином, при використанні першої форми, циклічно виконується оператор, що йде за ключовим словом **`while`** доти, поки виконується умова, яка знаходиться у круглих дужках. При використанні другої форми, циклічно виконується оператор, що знаходиться за ключовим словом **`do`** доти, поки виконується умова, яка знаходиться у круглих дужках. Розходження двох форм у послідовності виконання оператора і перевірки умови. Логіка виконання циклу **`while`** для першої форми приведена на рис. 4.2.

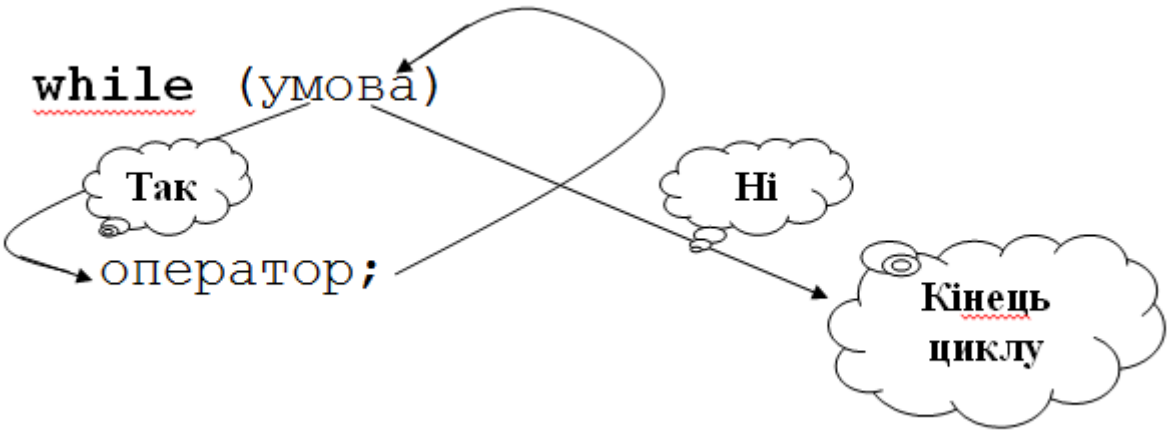


Рис. 4.2. Логіка виконання циклу **while**

Приклади:

а) **i=5; b=1;**

```
while(i > 0) { i--; b*=a; }
```

У результаті змінна **b** буде дорівнювати 5-ому ступеню змінної **a**.

б) **i=5; b=1;**

```
do { i--; b*=a; } while (i > 0);
```

У результаті змінна **b** буде дорівнювати 6-ому ступеню змінної **a**.

в) **while(1) оператор**.

Оператор буде виконуватися нескінченне число разів.

Як вправу розпишіть по тактах виконання перших двох прикладів.

4.6. Ключові слова break і continue

Оператор слово **break** перериває виконання операторів циклів **while** і **for**.

Ключове слово **continue** передає керування на початок циклів **while** або **for**. У перекладі з англійського break означає "перервати", continue - "продовжити".

Приклади:

а)

```
a=1; n=10;
for(i=0; i < n; i++)
```

```
  if(a >= 100)
    break;
  else
    a*=5;
```

Розпишемо по кроках виконання цього прикладу:

1-ий крок: **i=0; 0 < 10** - вірно, **1 >= 100** - не вірно, **a=1*5=5**, **i=0+1=1**;

2-ий крок: **1 < 10** - вірно, **5 >= 100**- не вірно, **a=5*5=25**, **i=1+1=2;**

3-ій крок: **2 < 10** - вірно, **25 >= 100** - не вірно, **a=25*5=125, i=2+1=3;**

4-ий крок: **3 < 10** - вірно, **125 >= 100** - вірно, керування передається операторові **break**, що перериває виконання циклу **for**.

б)

```
a=6; n=4;
for(i=0; i < n; i++)
  { a-=2;
    if(a != 0)
      b=1/a;
    else
      continue;
  }
```

Виконання цього прикладу по кроках буде виглядати так:

1-ий крок: **i=0; 0 < 4** – вірно, **a=6-2=4, 4 != 0** – вірно, **b=1/6=0.1667, i=0+1=1;**

2-ий крок: **1 < 4** – вірно, **a=4-2=2, 2 != 0** – вірно, **b=1/4=0.25, i=1+1=2;**

3-ій крок: **2 < 4** – вірно, **a=2-2=0, 0 != 0** – не вірно, керування передається операторові **continue**, що повертає керування на початок циклу, тобто на збільшення лічильника циклу **i=2+1=3;**

4-ий крок: **3 < 4** - вірно, **a=0-2=-2, -2 != 0** - вірно,

b=1/(-2)=-0.5, i=3+1=4;

5-ий крок: **4 < 4** – не вірно, кінець циклу.

5. ОСНОВНІ ПРАВИЛА СКЛАДАННЯ ПРОГРАМ

5.1. Підключення необхідних бібліотек мови Сі

Підключення стандартних бібліотек виконується на самому початку програми розміщенням наступного рядка:

```
#include <назва бібліотеки>
```

У перекладі з англійської “include” означає "підключити".

Кожна бібліотека підключається окремим рядком. Необхідність підключення тієї або іншої бібліотеки визначається використовуваними у програмі стандартними функціями, наприклад, функції, які виводять інформацію на екран, в основному, розташовано у бібліотеці **stdio.h**, математичні функції – у бібліотеці **math.h**, і т.д. Для того, щоб довідатися, у якій бібліотеці знаходиться функція, необхідно знаходячись у компіляторі мови Сі, підвести курсор до імені цієї функції і натисканням **F1** викликати для неї підказку (help).

5.2. Ключове слово main()

“Main” у перекладі з англійської означає “основний”. У мові Сі вводять поняття основної функції **main()**, призначеної для об'єднання інших стандартних і користувацьких функцій до файлу, що виконується. До введення поняття користувацької функції, усі наші програми будуть оформлені у вигляді функції main у такий спосіб:

```
void main ()
{ текст програми },
```

тобто після слова **main** відкривають і закривають круглі дужки і далі починається текст програми, обов'язково заключений у фігурні дужки. Ключове слово **void** переводять як «порожній», це означає, що програма нічого не повертає операційній системі.

5.3. Опис змінних

Усі використовувані у програмі змінні повинні бути описані на початку програми, тобто визначено їхній тип. Після ключового слова даного

типу містяться через кому імена змінних цього типу, використовувані у програмі, змінні одного типу можуть бути оголошені декількома групами. Перерахування змінних одного типу завжди завершується крапкою з комою.

Приклад:

```
int k, l, m;
int i, j;
float s, v;
unsigned char name1, name2.
```

5.4. Реалізація основних математичних функцій мови Сі

Прототипи всіх математичні функцій знаходяться у бібліотеці **math.h.**

Нижче наведено відповідність основних математичних функцій функціям, реалізованим мовою Сі:

$\sqrt{x}$ ~ **sqrt(x);**

$|x|$ ~ **abs(x)** для цілих чисел (**char, int**) і **fabs(x)** для чисел з плаваючою комою, **(float, double**);

$[x] = entier(x)$ ~ **floor(x)**, реалізована також функція **ceil(x)**, що повертає мінімальне ціле число, що перевершує **x**;

a^x ~ **pow(a,x)**;

e^x ~ **exp(x)**;

$\sin x$ ~ **sin(x)**;

$\cos x$ ~ **cos(x)**;

$tg\ x$ ~ **tan(x)**;

$\arcsin x$ ~ **asin(x)**;

$\arccos x$ ~ **acos(x)**;

$arctg\ x$ ~ **atan(x)**;

$\ln x$ ~ **log(x)**;

$\lg x$ ~ **log10(x)**;

Як бачимо, у мові Сі не реалізовано функцію обчислення логарифму по заданій основі. Для обчислення логарифму довільної основи необхідно

скористатися формулою переходу від однієї основи до іншої: $\log_b a = \frac{\log_c a}{\log_c b}$.

У цій же бібліотеці визначено деякі математичні константи, зокрема числа π і e з точністю до 20 знаків після коми, позначені **`M_PI`** і **`M_E`**.

Відзначимо, що, як і у багатьох інших мовах програмування, аргументи функції беруть до круглих дужок, поділ декількох аргументів виконують за допомогою коми. Аргументи усіх прямих тригонометричних функцій задають у радіанах, відповідно, зворотні тригонометричні функції повертають значення кута також у радіанах.

У якості прикладу розглянемо реалізацію мовою Сі функції, яку задано на трьох інтервалах. На початку розглянемо задачу у загальному вигляді:

$$f(x) = \begin{cases} f_1(x), \text{ якщо } x < a, \\ f_2(x), \text{ якщо } x \in [a,b], \\ f_3(x), \text{ якщо } x > b. \end{cases}$$

Графічна інтерпретація функції, заданої на інтервалах, наведено на рис. 5.1.

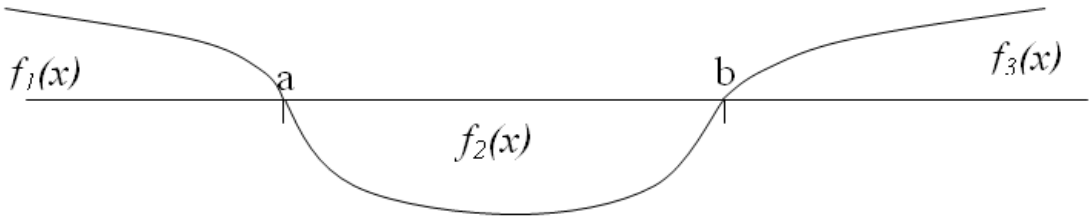


Рис. 5.1. Графічна інтерпретація функції, заданої на інтервалах

На початку наведемо схему обчислення функції ***f(x)*** (рис. 5.2):

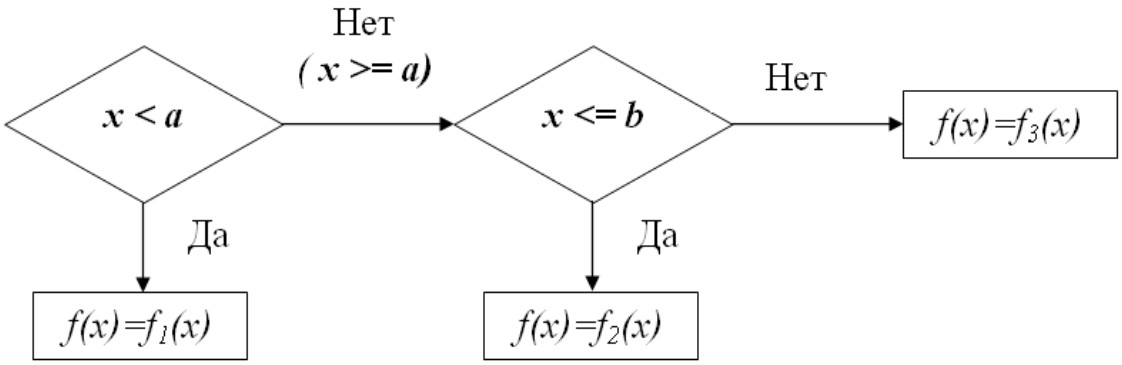


Рис. 5.2. Блок-схема обчислення значень функції, заданої на трьох інтервалах

Ясно, що через оператор умови ***if*** обчислення можна записати так:

```
if(x < a)
  f=f1(x);
else
if(x <= b)
    f=f2(x);
else
    f=f3(x);
```

Замість функцій ***f1(x), f2(x), f3(x)*** необхідно написати конкретні формули для обчислень. Наведемо тепер фрагмент програми для реалізації конкретної функції, яка задана на ітервалах:

$$f(x)=\begin{cases}\dfrac{Sin^4 x+Ctg^4(x+1)}{\sqrt[5]{1+|Cos\ x|}}, \text{если } x\le -\dfrac{\pi}{2},\\ \log_2(x+\dfrac{\pi}{2})+\log_{\sqrt{2}}(x+\pi), \text{если } x\in(-\dfrac{\pi}{2},\dfrac{\pi}{2}),\\ \dfrac{1}{\sqrt{2\pi}}e^{-\left(\frac{x-1}{2}\right)^2}, \text{если } x>=\dfrac{\pi}{2}.\end{cases}$$

```
if(x <= -M_PI/2)
  f=(pow(sin(x),4)+1/pow(tan(x+1),4))/(pow(1+
  fabs(cos(x)),1.0/5));
else
if(x < M_PI/2)
```

```
    f=log(x+M_PI/2)/log(2)+log(x+M_PI)/log(sqrt(2));
else
    f=1/sqrt(2*M_PI)*exp(-pow((x-1)/2,2));
```

Зауваження. Мовою Сі обчислення з цілими числами виконуються у класі цілих чисел, тому, наприклад, результатом виконання оператора 1/5 буде 0. Якщо ж хоч одна змінна в обчислюваному виразі є змінною з плаваючою комою, то усі обчислення виконуватимуться у класі чисел з плаваючою комою. Тому у виразі для першої функції дріб 1/5 необхідно записати як 1.0/5 (можна записати 1./5).

6. МАСИВИ

У математиці та інших науках широко використовують змінні з індексами (вектори, матриці, тензори тощо). Для їх реалізації у мовах програмування введено поняття масиву.

6.1. Опис масивів

Масиви описують, як і звичайні змінні, тільки після імені масиву у квадратних дужках указують довжину кожної розмірності масиву.

Приклади:

Рядок

```
int a[10];
```

описує масив із десяти цілих чисел: `a[0],...,a[9]`.

Зауваження. Мовою Сі нумерація елементів масивів починається з нуля.

Рядок

```
float a[3][3];
```

описує двовимірний масив чисел з плаваючою комою (матрицю) ***3 x 3***. Як і у математиці, перший індекс – це номер рядку, другий індекс – номер стовпця. Відповідність математичного запису матриці запису мовою Сі наступне:

$$\begin{matrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{matrix} \Rightarrow \begin{matrix} a[0][0] & a[0][1] & a[0][2] \\ a[1][0] & a[1][1] & a[1][2] \\ a[2][0] & a[2][1] & a[2][2] \end{matrix}$$

Рядок

```
float f[5][6][7][9];
```

описує чотирьохвимірний масив чисел з плаваючою комою, із загальним числом елементів**5*6*7*9=1890**.

6.2. Конструкція #define для оголошення розмірності масивів.

Оскільки довжину масиву приходиться використовувати в багатьох місцях програми, вкрай незручно явно вказувати довжину, як приведено в розділі 6.1. Справа в тому, що для зміни довжини масиву прийдеться коректувати програму в багатьох місцях. Для того, щоб уникнути цього, необхідно довжину розмірності масиву оголосити через конструкцію **#define**, що міститься до ключового слова **main**, як правило, після підключення бібліотек. Після ключового слова **#define**має бути пробіл, потім ім'я константи, і через пробіл значення, що привласнюється. Ніякі інші знаки не ставляться. Конструкція **#define** може використовуватися для оголошення будь-яких типів констант.

Нижче приводиться приклад програми ініціалізації матриці за заданою формулою:

```
#define N 3
#define M 3
void main()
{ float a[N][M];
  int i,j;
  for(i=0; i < N; i++)
    for(j=0; j < M; j++)
      a[i][j]=pow(i+1,j);
}
```

Детально розглянемо роботу наведеного подвійного циклу **for.** Передусім відповідно до синтаксису оператора, наведеного у розділі 4.3, ви-

значимо складові елементи зовнішнього циклу **for** по змінній **i**. Результат графічно зображено на рис. 6.1.

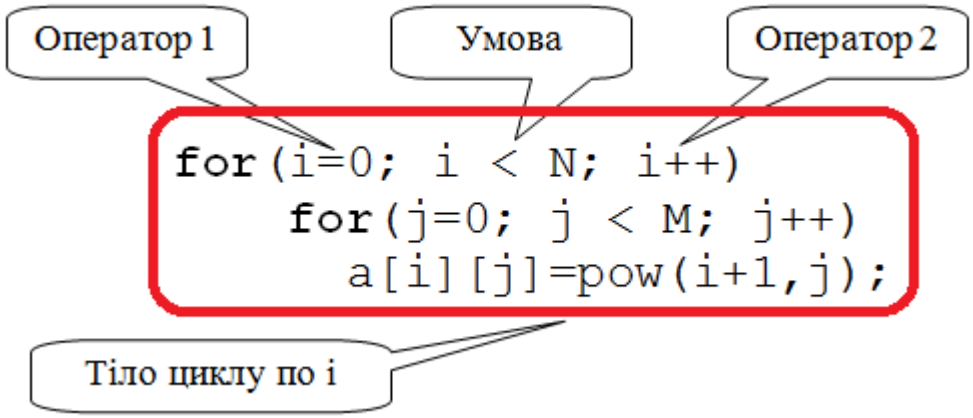


Рис. 6.1. Елементи зовнішнього циклу for

Складові внутрішнього циклу по змінній ***j*** очевидні. Покрокове виконання подвійного циклу буде наступним:

1 крок: i=0, 0 < 3 - так, j=0, 0 < 3 - так, a[0][0]=pow(1,0)=1;
2 крок: j=1, 1 < 3 - так, a[0][1]=pow(1,1)=1;
3 крок: j=2, 2 < 3 - так, a[0][2]=pow(1,2)=1;
4 крок: j=3, 3 < 3 - ні, кінець циклу по j;
5 крок: i=1, 1 < 3 - так, j=0, 0 < 3 - так, a[1][0]=pow(2,0)=1;
6 крок: j=1, 1 < 3 - так, a[1][1]=pow(2,1)=2;
7 крок: j=2, 2 < 3 - так, a[1][2]=pow(2,2)=4;
8 крок: j=3, 3 < 3 - ні, кінець циклу по j;
9 крок: i=2, 2 < 3 - так, j=0, 0 < 3 - так, a[2][0]=pow(3,0)=1;
10 крок: j=1, 1 < 3 - так, a[2][1]=pow(3,1)=3;
11 крок: j=2, 2 < 3 - так, a[2][2]=pow(3,2)=9;
12 крок: j=3, 3 < 3 - ні, кінець циклу по j;
13 крок: i=3, 3 < 3 - ні, кінець циклу по i.

В результаті ми отримали наступну матрицю

$$a = \begin{pmatrix} 1 & 1 & 1 \\ 1 & 2 & 4 \\ 1 & 3 & 9 \end{pmatrix}$$

6.3. Ініціалізація масивів при оголошенні

Масивові можна привласнити конкретні значення безпосередньо при його оголошенні (описі). Для цього необхідно у фігурних дужках через кому перелічити значення, що привласнюють, наприклад:

```
int a[5]={0,1,2,3,4};
float a[3][2]={ {1.8, 2.4},
                {3.2, 4.9},
                {5.5, 6.9}};
```

7. ФУНКЦІЯ PRINTF()

Функція **printf()** - одна з функцій мови Сі, яка призначена для виведення форматованої інформації на екран.

7.1. Виведення на екран заданого тексту

Будь-який текст, узятий у лапки і поміщений усередині круглих дужок, як аргумент функції **printf()**, без змін виводиться на екран.

Приклади.

```
printf("Good morning!");
printf("Добрий ранок!");
printf("a=b");
```

Якщо усередині лапок поставити два символи **\n**(разом!), то їх надруковано не буде, а текст після цих знаків буде виведено з нового рядку. Цю пару часто називають символом переведення каретки. Наприклад, рядок

```
printf("Рішення задачі:\na=34.");
```

виведе на екран:

```
Рішення задачі:
a=34.
```

Зауваження. Перший аргумент функції printf(), поміщений у лапки, слід розміщувати в один рядок.

7.2. Виведення на екран значень змінних

Для того, щоб вивести на екран значення змінної, необхідно у першому аргументі функції **printf()**, що завжди беруть у лапки, помістити знак відсотку **(%)** і за ним формат виведеної змінної, а другим аргументом помістити ім'я цієї змінної. Одним звертанням до функції **printf()** можна вивести на дисплей значення декількох змінних. Для цього знак

відсотку з форматом указують стільки разів, скільки самих змінних, а їхні імена перелічують через кому після першого аргументу.

Основні типи форматів, використовувані у функції **printf()**:

c - одиночний символ (char), виводиться символ, що відповідає коду цього числа у таблиці ASCII (таблиця відповідності кодів і символів).

d – цілі числа (int, char),

f – числа з плаваючою комою (float, double),

s – рядок символів (масив char).

Число перед форматом для форматів **d** означає кількість позицій, що відводиться для виведення числа, що дозволяє проводити необхідне вирівнювання виведених чисел. Наприклад, якщо значення трьох змінних **x1, x2, x3** рівні відповідно **5, 425 і 28,** то рядок програми **printf("%3d\n%3d\n%3d.\n", x1,x2,x3)** виведе на екран:

```
  5
425
 28.
```

Якщо формат **d** указати без 3, то на екран буде виведено:

```
5
425
28.
```

Формат **f** може бути задано у вигляді **%k.nf,** де **k** - загальна кількість позицій у числі, включаючи крапку, що розділяє цілу і дробову частини числа, **n** – кількість позицій, що відводять для знаків після крапки. Допускається формат без указування **k**, тобто **%.nf.**

Приклад:

```
x1=2.375; x2=31.697; x3=5.1;
printf("x1=%5.2f, x2=%5.2f, x3=%5.2f.",x1,x2,x3);
```

На екран буде виведено рядок:

```
x1= 2.38, x2=31.70, x3= 5.10.
```

Зверніть увагу, що при відкиданні десяткових знаків виконується округлення.

8. ПРИКЛАДИ ПРОГРАМ

8.1. Друк координат вектора, які перевищують заданий поріг

```
#include <stdio.h>
#define N 5
main()
{ int i;
  float c;
  float a[N]={17.3,8.45,10.0,15.23,5.01};

  c=10;
  for(i=0; i<N; i++)
    if(a[i] >= c)
      printf("a[%d]=%.3f ",i,a[i]);
  printf("\n");
}
```

При заданих значеннях змінних програма виводить на екран рядок:
a[0]=17.300 a[2]=10.000 a[3]=15.230.

Переконайтеся у цьому, розписавши роботу програми по тактах і запустивши її на виконання після компіляції.

8.2. Множення двох прямокутних матриць

Нагадаємо, що операція множення вводиться до математики для двох прямокутних матриць, у яких кількість стовпців першої матриці збігається з кількістю рядків другої матриці, кількість рядків результуючої матриці дорівнює кількості рядків першої матриці, кількість стовпців – кількості стовпців другої матриці:
A(m x n)*B(n x k)=C(m x k).

Елементи матриці С обчислюють за наступною формулою:

$$c_{ij} = \sum_{l=1}^{n} a_{il} b_{lj} (i = 1...m, j = 1...k).$$

```
#include <stdio.h>
#define M 3
#define N 2
#define K 4
void main()
{ int i,j,l;
  int a[M][N]={{1,2},{3,4},{5,6}};
  int b[N][K]={{7,8,9,10},{11,12,13,14}};
  static int c[M][K];

  for(i=0; i<M; i++)
    { for(j=0; j<K; j++)
        { for(l=0;l<N;l++)
            c[i][j]+=a[i][l]*b[l][j];
          printf("c[%d][%d]=%3d  ",i,j,c[i][j]);
        }
      printf("\n");
    }
}
```

При заданих значеннях змінних програма виводить на екран:

```
c[0][0]= 29 c[0][1]= 32 c[0][2]= 35 c[0][3]= 38
c[1][0]= 65 c[1][1]= 72 c[1][2]= 79 c[1][3]= 86
c[2][0]=101 c[2][1]=112 c[2][2]=123 c[2][3]=134.
```

8.4. Розкладення числа на прості множники

```
#include <stdio.h>
#define N 1000
#define M 50

void main()
{ unsigned int p[N]={2,3,5,7,11,13,17,19,23,...};
  int i,k;
  unsigned int pn[M], ms[M];
  char mr, prd;
  unsigned long n;
```

```
  n=1617;
  if(n/2 > p[N-1])
    { printf("Необхідно збільшити масив простих чи-
сел!");
      return;
    }
  k=0;
  printf("%li=",n);
  for(i=0; i<N && n>=p[i]; i++)
    { mr=prd=0;
      while(n%p[i]==0)
        { n/=p[i];
          mr++;
          prd=1;
        }
      if(prd==1)
        { pn[k]=p[i];
          ms[k]=mr;
          k++;
        }
    }
  for(i=0; i<k; i++)
    printf("%d(%d)",pn[i],ms[i]);
}
```

При заданих значеннях змінних програма виводить на екран:
1617=3(1)7(2)11(1), що еквівалентно розкладанню
$1617=3*7^2*11$.

Масив простих чисел **p[N]** повинний бути заповнений до необхідного обсягу. Ключове слово **return** завершує програму.

8.5. Рішення фізичної задачі

Задача. Температура T_M молока у бідонах після змісту їх на відкритому повітрі в плині t годин виражається наступною формулою:

$$T_м(t) = T_B + (T_м(0) - T_B)\exp\left(-\frac{kSt}{V}\right),$$

де T_B - температура навколишнього повітря в ^oC, $T_M(0)$ - початкова температура молока в ^oC, S - площа поверхні бідона в м^2, V - об'єм поверхні бідона в м^3, k =0.00448 м/год. – постійний коефіцієнт. Написати програму обчислення температури молока після збереження, що забезпечує видачу результатів у вигляді таблиці з пояснювальним текстом для t від 0 до $t_{\max}$ з кроком Δt.

До початку написання програми необхідно надати імена змінним, які будуть використані у програмі. Наполегливо рекомендуємо, по можливості, зберегти позначення, прийняті у формулюванні задачі. Для позначень у задачі, що не можуть служити іменами змінних мовою Сі (див. визначення імені змінної з розділу 1), бажано надати імена, пов'язані із початковими позначеннями. Для розглянутої задачі неприпустимими іменами є: T_M, T_B, $T_M(0)$, $t_{\max}$, Δt (використання не латинських літер і дужок). Надамо їм наступні, дозволені імена: **Tm, Tv, Tm0, tmax, dt**. Оскільки усі ці величини можуть бути не цілими числами, оголосимо їх як **float**. Для циклічного обчислення температури зручніше за все скористатися оператором циклу **for.**

```
#include <stdio.h> // бібліотека функцій введення-виведення
#include <math.h>// бібліотека математичних функцій
#include <conio.h> // бібліотека функцій для реакції на
//натиснення клавіш клавіатури
#define k 0.0048
void main()
{ float Tm,Tv,Tm0,S,V,dt,tmax,t;
/* Введення до програми значень вхідних даних найпростішим способом – оператором присвоєння: */
```

```
 Tv = -5;  // температура навколишнього повітря у
градусах Цельсію
 Tm0 = 36; // початкова температура молока у градусах
Цельсію
 S = 1.2;  // площа поверхні бідона у м. кв.
 V = 0.05; // обсяг бідона у м. куб.
 tmax = 8; // інтервал розрахунку у годинах
 dt = 0.5; // крок розрахунку у годинах
// Друк заголовку таблиці:
 printf("Час (години) Температура молока у град. Це-
льсію\n");
 for(t=0;t <= tmax;t+=dt)
   { Tm=Tv+(Tm0-Tv)*exp(-k*S*t/V);
     printf(" %.1f %.2f\n",t,Tm);
   }
getch(); // функция припиняє роботу програми до нати-
скання
        // будь-якої клавіші
}
```

У цій програмі використано ще одну конструкцію мови Сі – коментар.

Визначення: Коментар – це пояснювальний текст, вставлений до програми, який не аналізується компілятором.

Є два способи перетворення тексту до коментаря. Весь текст, поміщений до дужок **/*…*/**і текст, що йде за знаками **//** до кінця рядка, є коментарем. Так само виділяють тимчасово не використовуваний код програми.

9. РОБОТА У ІНТЕГРОВАНОМУ СЕРЕДОВИЩІ ПРОГРАМУВАННЯ C++Builder6

9.1. Запуск програми та вибір проекту

Для запуску програми можна скористатися кнопкою «Пуск»: **Пуск=>Все программы=>Borland C++ Builder 6=>C++ Builder 6.**

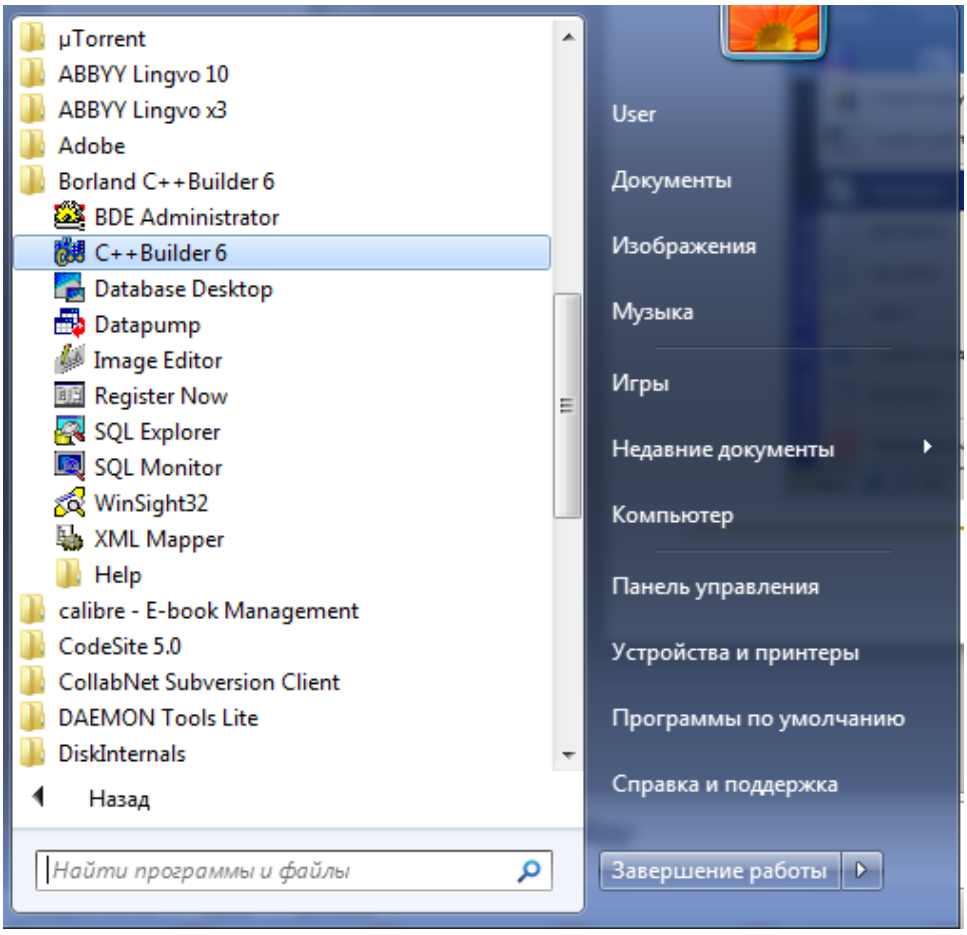


Рис. 9.1. Запуск C++ Builder 6

Після запуску програми за замовчуванням завантажується шаблон для типу проекту «Application». Ми будемо використовувати найбільш простий тип проекту «Console Application». Для його відкриття необхідно виконати **File =>New =>Other =>Console Wizard** (майстер консольних додатків).

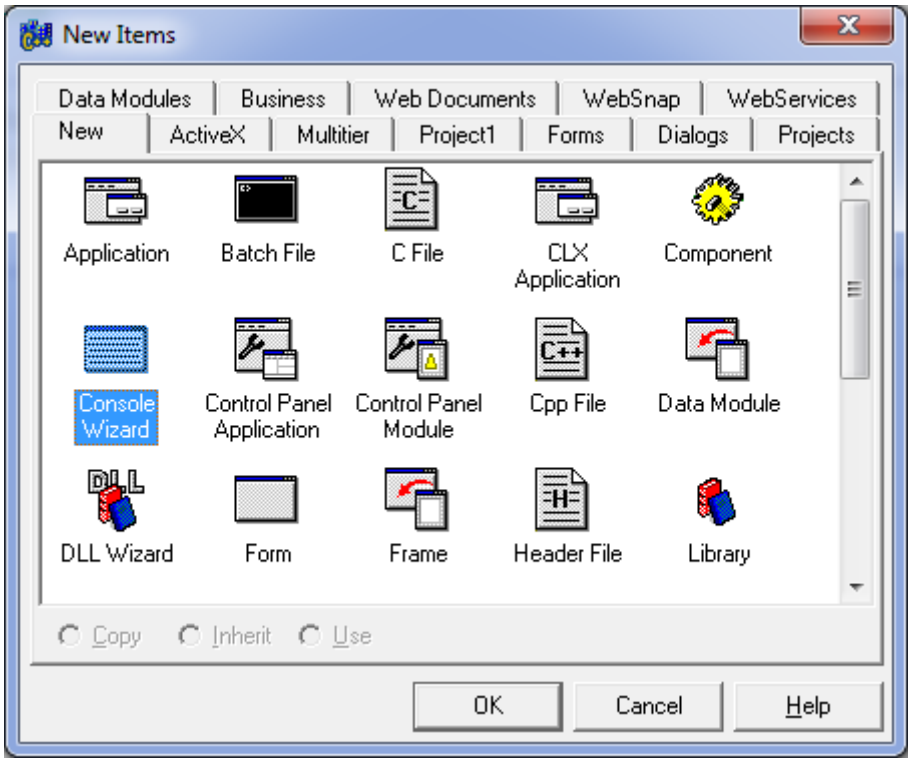


Рис. 9.2. Вікно вибору типу проекту

У результаті буде відкрито вікно вибору параметрів проекту, у якому необхідно зробити налаштування, які вказано на рис. 9.3:

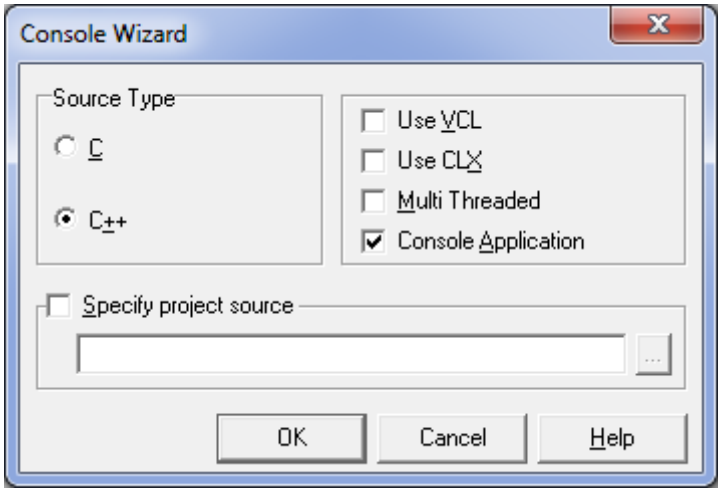


Рис. 9.3. Вікно вибору параметрів проекту

Закінчиться установка проекту відкриттям вікна для введення тексту програми за шаблоном для основної функції **`main():`**

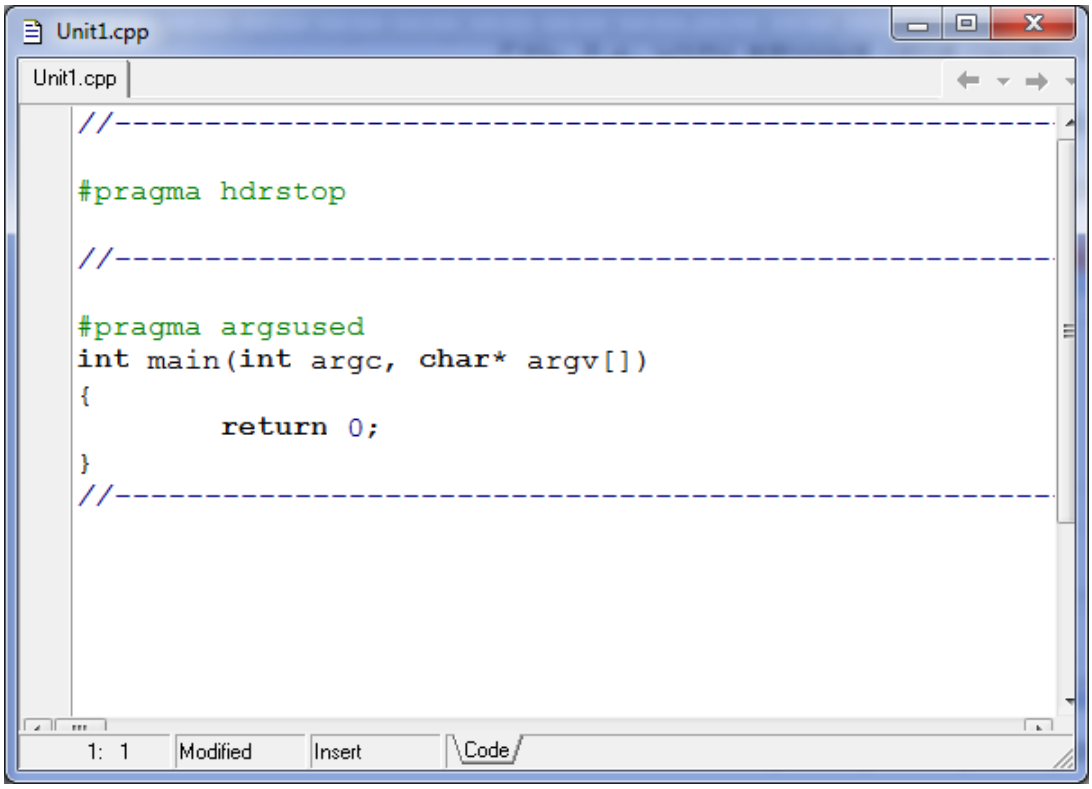


Рис. 9.4. Вікно для введення тексту програми

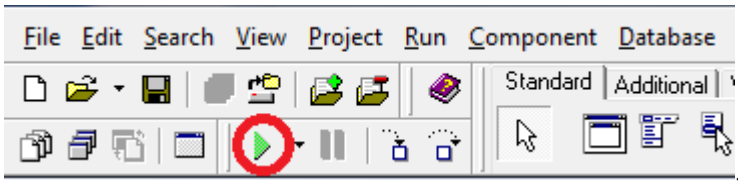
Новостворений проект обов'язково потрібно зберегти в окремому каталозі за допомогою команди **File => Save All** або скористатися значком на панелі інструментів: 

.

При цьому проекту слід надати ім'я, пов'язане із змістом розв'язуваної задачі, обов'язково англійськими літерами без проміжків. Із запропонованим ім'ям **Unit1.cpp** для файлу з текстом програми можна погодитися. Якщо проекту, наприклад, дати ім'я **SilosnBash**, то в обраному каталозі з'являться такі файли:

- **SilosnBash.bpf** – файл з типом проекту;
- **SilosnBash.bpr** – основний файл проекту з його параметрами, його можна відкривати за допомогою файлового менеджера для завантаження проекту;
- **SilosnBash.res –** файл з ресурсами проекту;
- **Unit1.cpp** – файл з текстом програми (ім'я файлу не пов'язане з ім'ям проекту та може бути довільним).

9.2. Компіляція і налагодження програми

Після завершення введення тексту програми, можна її відправити на компіляцію командами **Run=>Run або,** скориставшись значком на панелі інструментів: 

, або натисканням клавіші **F9.** При цьому, насамперед, буде проведене перевірку тексту на наявність синтаксичних помилок, тобто таких помилок, які не дозволяють перекласти текст програми машинними кодами. Приклад завершення компіляції програми з виявленням помилок наведено на рис. 9.5.

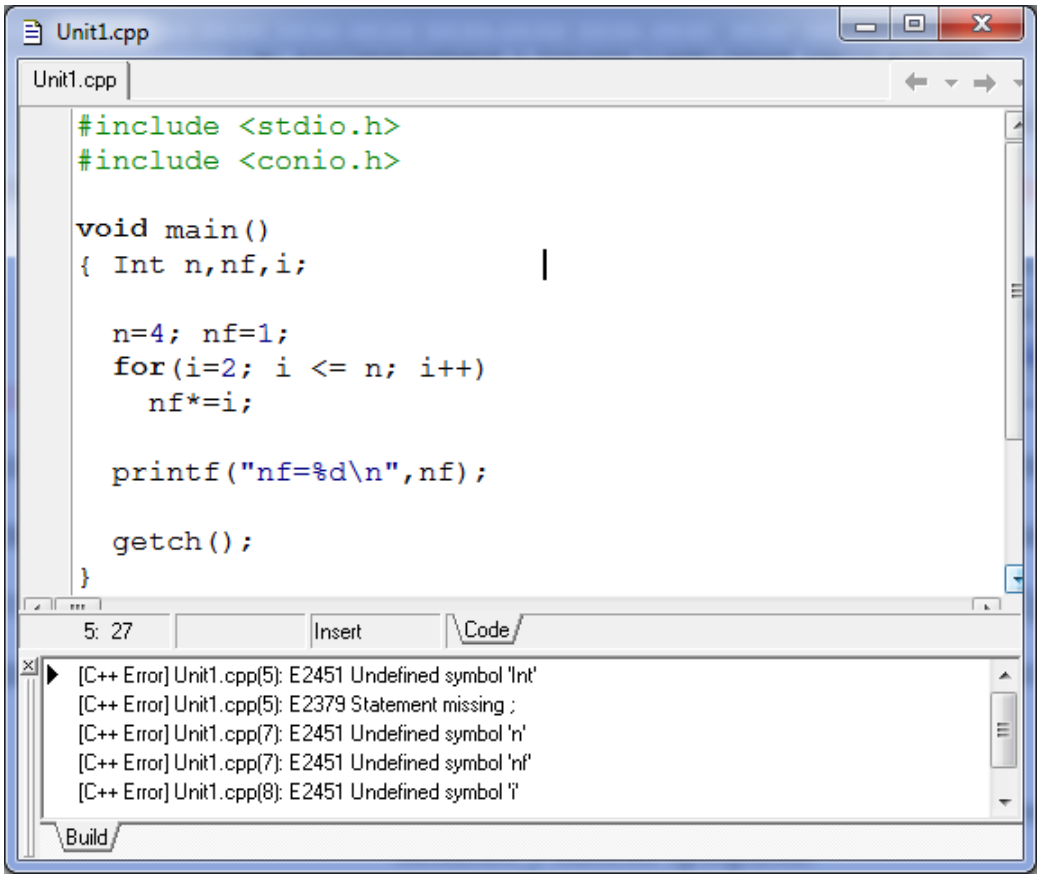


Рис. 9.5. Вигляд вікна компілятора у випадку виявлення помилок у тексті програми

Після виправлення першої помилки, рекомендується повторити компіляцію, оскільки інші помилки можуть бути наслідком першої, як це і сталося в програмі, введеної до компілятора в якості прикладу. Ключове слово **`int`** було написано з великої літери, трактоване як ім'я змінної, відповідно, перелічені змінні виявилися не описаними. Зверніть увагу, що

ключові слова мови компілятор автоматично відокремлює жирним шрифтом. Компілятор також може видати попередження (**warnings**) про некоректну роботу програми:

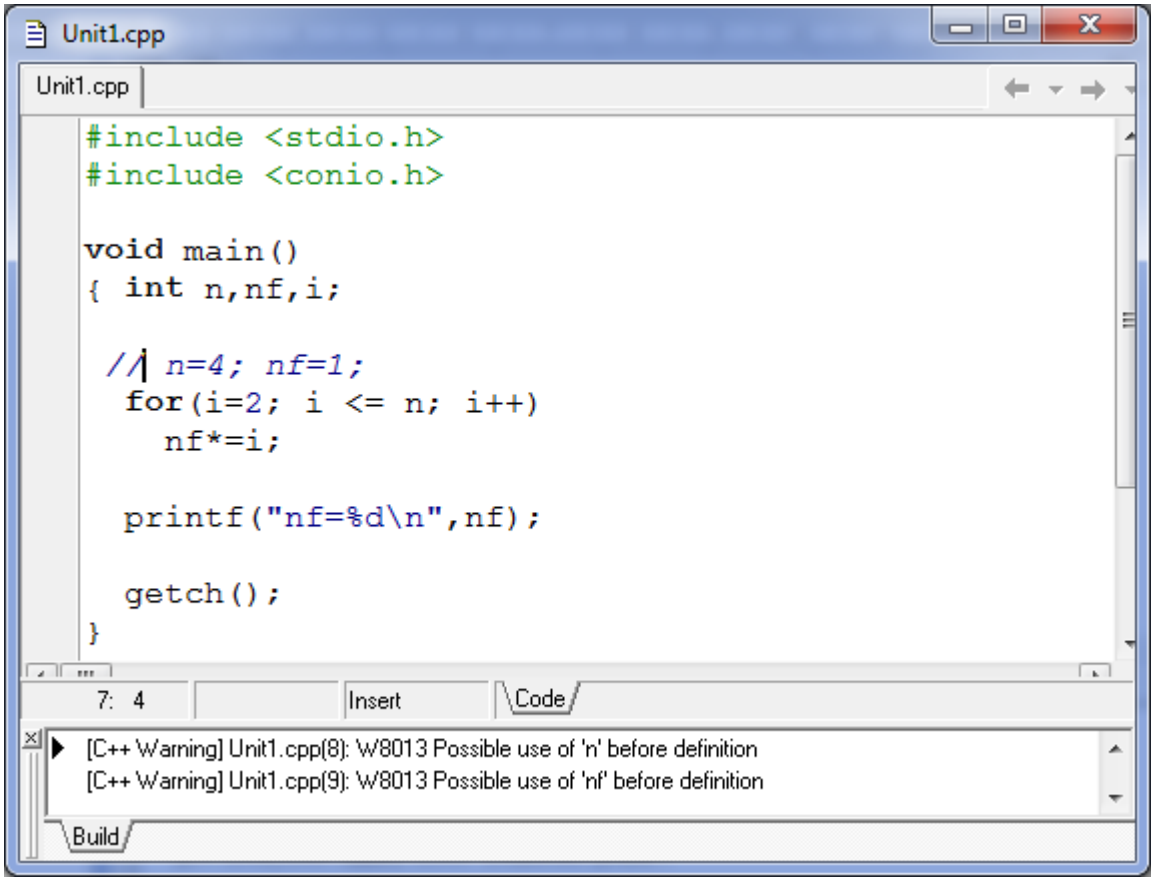


Рис. 9.4. Вигляд вікна компілятора з попередженнями

У даному випадку компілятор попереджає, що обидві змінні використовуються без присвоєння їм початкових значень. Наполегливо рекомендуємо такі місця у програмі теж виправляти.

За умови успішної компіляції, у каталозі проекту буде створено файли **SilosnBash.obj** і **SilosnBash.exe** (основна частина імені цих файлів збігається з ім'ям проекту). Перший з них містить машинний код програми, яка компілюється, другий є файлом, готовим до виконання, із усіма необхідними атрибутами.

9.3. Використання режиму налагодження (debugging)

Зауважимо, що успішна компіляція програми ні в якому разі не гарантує її правильну роботу. Необхідно перевірити роботу програми, принай-

мні, на двох контрольних варіантах початкових даних, у яких результати роботи програми отримано способом, незалежним від самої програми, наприклад, виконано на калькуляторі, або у середовищі MS Excel.

Для налагодження програми, яка пройшла компіляцію, зручно використовувати режим налагодження, який дозволяє зупинити програму у заданому місці і виконувати її по кроках із можливістю відслідковувати значення змінних у процесі виконання. Для роботи у режимі налагодження необхідно на потрібний рядок у програмі поставити переривання, натиснувши лівою кнопкою мишки на сірому полі ліворуч від тексту або натиснувши клавішу **F5**. Після запуску програма зупиниться **перед** виконанням вказаного рядку, як показано на рис. 9.7:

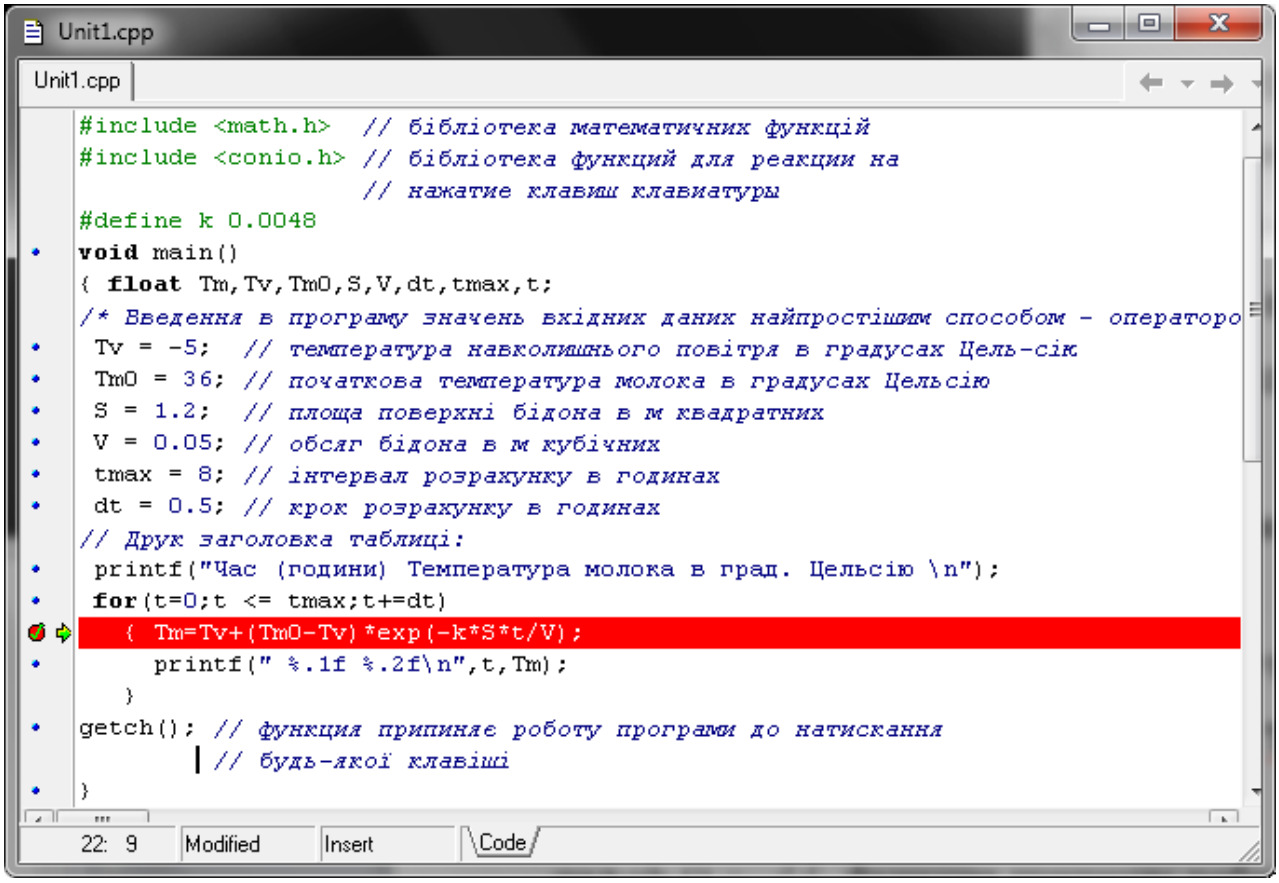


Рис. 9.7. Установка переривання у програмі

Для перегляду значень змінних достатньо підвести курсор миші до імені змінної або поставивши курсор на ім'я змінної і натиснути **Alt+F5**, як показано на рис. 9.8:

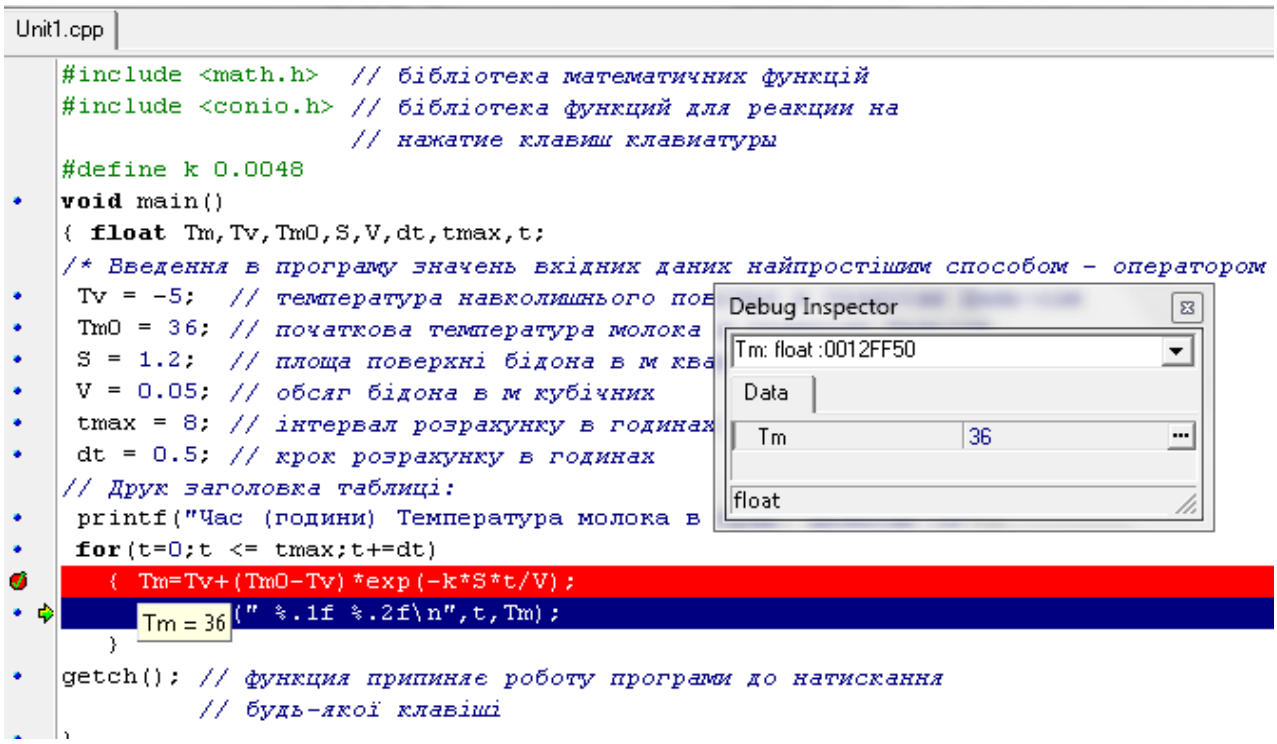


Рис. 9.8. Перегляд значень змінних

Для відрядкового виконання програми використовується клавіша **F8**.

10. ВВЕДЕННЯ ІНФОРМАЦІЇ ДО ПРОГРАМИ З ЕКРАНУ

10.1. Функція scanf()

Функція **`scanf()`** є зворотною до функції **`printf()`**, тобто забезпечує введення даних з екрану.

Правило написання аргументів функцій scanf() і printf() також багато у чому збігаються. У першому аргументі, що завжди наводять у лапках, після знаку відсотку (%) указують формат введення змінної, а у другому аргументі не саме ім'я змінної, а її адресу. Для використання адреси змінної перед її ім'ям необхідно поставити знак амперсанду **`&`**. Одним викликом функції **`scanf()`** можна ввести значення декількох змінних.

Типи форматів функції scanf() збігаються з типами форматів, наведеними для функції **`printf()`** у розділі 7.2 за одним виключенням – при введенні чисел з плаваючою комою, з подвійною точністю (**`тип`** double), необхідно використовувати **`формат`** lf.

Приклади:

```
float v;
```

```
double t;
scanf("%f%lf", &v,&t);

int day, year;
char month[20];
scanf("%d%s%d",&day,month,&year);
```

У цьому випадку перед ім'ям змінної **month** не поставлено знак **&**, у зв'язку із тим, що для масивів ім'ям масиву є адреса першого елемента масиву, тобто **month=&month[0]**.

Виклик функції **scanf()** приводить до зупинки програми, і користувач одержує можливість набору інформації на екрані. Після натискання клавіші "Enter", здійснюється спроба присвоєння набраних на екрані значень зазначеним змінним. Присвоєння завершується успішно, якщо формат у функції **scanf()** і набрана на екрані інформація відповідають типам змінних, які вводять. Функція **scanf()** повертає кількість успішно введених значень.

10.2. Приклади програм, що використовують введення даних з екрану

10.2.1. Введення вектора з екрану.

```
#include<stdio.h.>
#define N 1000
void main()
{ int a[N],n,i;
  printf("Введіть розмірність вектора:\n");
  printf("(не більш %d)\n",N);
  scanf("%d",&n);
  if(n > N)
   { printf("Не припустима розмірність! \n");
     return;
   }
 for(i=0; i<n; i++)
   { printf("Введіть %d координату вектора:\n",i+1);
     scanf("%d",&a[i]);
```

```
    }
}
```

У цій програмі з'являється розмірність масиву із деяким запасом, максимальна розмірність масиву доводиться до користувача, якщо все-таки користувач уводить велику розмірність, програма завершується виведенням на екран аварійного повідомлення. У цьому випадку подальша робота програми є непередбачуваною, через можливу зміну оперативної пам'яті, не виділеної для програми, у тому числі у системній області. Найчастіше у таких ситуаціях операційна система виводить повідомлення «Програма виконала неприпустиму операцію, її буде закрито». У випадку введення припустимого значення розмірності, програма послідовно запитує зазначену кількість координат вектора, нумеруючи їх з 1.

10.2.2. Введення дати з екрану

```
#include <stdio.h>
#include <string.h>

void main()
{ int day=0, year=0,m=0,dmax,i;
  static char month0[10];
  char month[12][10]=
{{"січень"}, {"лютий"},{"березень"}, {"квітень"},
{"травень"}, {"червень"}, {"липень"}, {"серпень"},
{"вересень"}, {"жовтень"}, {"листопад"},{"грудень"}};
 char vvod=0;

 printf("Введіть рік:\n");
 while(vvod==0)
   { vvod=scanf("%d",&year);
     if(year <= 0 || year >= 5000)
       { printf("Рік введено неправильно! Повторіть
введення:\n");
         vvod=0;
       }
```

```
   }
 vvod=0;
 printf("Введіть місяць (повна українська на-
зва):\n");
 while(vvod == 0)
   { vvod=scanf("%s",month0);
     for(i=0;i<12&&stricmp(month0,month[i])!=0;i++);
     if(i == 12)
       { printf("Місяць набрано неправильно! Повто-
ріть введення:\n");
         vvod=0;
       }
     else m=i+1;
   }
 if((m <= 7 && m%2==1) || (m > 6 && m%2 == 0))
   dmax=31;
 else dmax=30;
 if(m == 2)
   if(year%4 == 0 && year%400 != 0 ) dmax=29;
   else dmax=28;
 vvod=0;
 printf("Введіть число місяця:\n");
 while(vvod == 0)
   { vvod=scanf("%d",&day);
     if(day <= 0 || day > dmax)
       { printf("Число набрано неправильно! Повторіть
введення:\n");
         if(day > dmax)
           if(dmax == 31)
             printf("(у місяці %s тільки %d день!)\n",
                    month0,dmax);
           else
             printf("(у місяці %s тільки %d днів!)\n",
                    month0,dmax);
         vvod=0;
```

```
        }
    }
 printf("Введено дату: %d %s %d.\n",day,month0,year);
}
```

Розмір цієї програми обумовлений необхідністю реалізації складного контролю даних, що вводяться до програми. Використана у програмі функція **stricmp()** забезпечує порівняння двох строкових змінних (строкова змінна – це масив символів, що закінчується нулем) і повертає не нульове значення, якщо рядки розрізняються.

10.2.3. Доопрацювання програми рішення фізичної задачі з розділу 8.5

Спочатку сформулюємо загальні вимоги до програми, що роблять її придатною до експлуатації:

1. Програма після запуску повинна виводити на екран назву задачі, яку розв'язують і прізвище виконавця;
2. Усі вхідні дані необхідно запитувати з повною назвою величини, що вводиться, і вводитися з клавіатури. При введенні фізичних величин необхідно вказувати її одиниці виміру;
3. Після друку результатів роботи програма повинна їх зберігати на екрані до натискання будь-якої клавіші.

```
#include <stdio.h>
#include <math.h>
#include <conio.h>
#define k 0.0048
void main()
{ float Tm,Tv,Tm0,S,V,dt,tmax,t;
 printf("Програма розрахунку зміни температури молока
\n");
 printf("у бідонах при утриманні на відкритому повіт-
рі. \n");
 printf("Розробила студентка групи Е11 Чалдомир Л.
\n");
```

```
 printf("Уведіть температуру навколишнього повітря в
гр. Цельсію \n");
 scanf("%f",&Tv);
 printf("Уведіть початкову температуру молока в гр.
Цельсію \n");
 scanf("%f",&Tm0);
 printf("Уведіть площу поверхні бідона в м квадратних
\n");
 scanf("%f",&S);
 printf("Введіть обсяг бідона в м кубічних \n");
 scanf("%f",&V);
 printf("Введіть інтервал розрахунку у годинах \n");
 scanf("%f",&tmax);
 printf("Уведіть крок розрахунку у годинах \n");
 scanf("%f",&dt);
// Друк заголовка таблиці:
 printf("Час (години) Температура молока в град. Це-
льсію\n");
 for(t=0;t <= tmax;t+=dt)
   { Tm=Tv+(Tm0-Tv)*exp(-k*S*t/V);
     printf(" %.1f       %.2f\n",t,Tm);
   }
 printf("Для виходу з програми натисніть будь-яку
клавішу. \n");
 getch();
}
```

Функція **getch()** із бібліотеки**<conio.h>**припиняє роботу програми до натискання будь-якої клавіші.

11. СТВОРЕННЯ ФУНКЦІЙ КОРИСТУВАЧА

Функції користувача вживаються для структурування програм. Без їхнього використання не може бути налагоджено будь-якої, хоч трохи складної програму.

11.1. Загальні правила написання функцій користувача

1. Тіло функції, тобто послідовність операторів, які виконуються функцією, міститься поза **main()** функції. Це може бути той же файл, де знаходиться **main()** функція, або інший файл, об'єднаний з файлом, що містить **main()** функцію, єдиним проектом.
2. Тіло функції складається з типу функції (тип змінної, яка повертається) її імені і типів аргументів у круглих дужках, розділених комами, і далі у фігурних дужках наводять оператори, виконувані функцією:

```
Тип функції Ім'я функції (тип арг1,тип арг2,..., тип аргn)
{ Опис змінних;
  Оператори;
}
```

   Якщо функція нічого не повертає, аналогічно при відсутності аргументів, використовується ключове слово **void.** До імені функції висувають ті ж вимоги, що і до імені змінної. Зрозуміло, не можна використовувати однакові імена для змінної і функції.
3. Якщо тіло функції розміщують після основної функції main(), перед main() необхідно помістити її прототип. Прототип функції збігається з першим рядком тіла функції і завершується крапкою з комою. Прототип використовується для перевірки правильності виклику функції.
4. Функцію можна викликати з будь-якого місця **main()** функції або у іншій функції користувача за допомогою рядку **Ім'я функції (арг1,арг2,…,аргn);**
5. Повернення значення функції забезпечується використанням у тілі функції ключового слова **return**.

11.2. Приклад використання функцій користувача

Розглянемо алгоритм чисельного рішення аналітично заданого рівняння. У загальному вигляді такий алгоритм не розглядають. Розглянемо випадок відшукання одного кореня рівняння **f(x)=0**, припускаючи, що інтервал, який містить тільки цей корінь нам відомий, функція безперервна і корінь утворено не торканням функцією осі **x,** а її перетином. Відому, за припущенням, інформацію можна легко відшукати на графіку функції. У цьому випадку можна застосувати метод інтервалів, який полягає у послідовному розподілі інтервалу навпіл з визначенням на кожнім кроці підінтервалу, який містить корінь. Рішення про потрібний інтервал може бути прийнято за знаком добутку $f(x_n)f(x_t)$, де x_n – початок інтервалу, а x_t - його середина. Якщо добуток негативний, корінь знаходиться у лівому підінтервалі, якщо позитивний – у правому, а якщо дорівнює нулю, то x_t – корінь (припускаючи, що кінці початкового інтервалу не є коренями).

```
#include<stdio.h>
#include<math.h>
#include<conio.h>
double f(double x); // Прототип функції

void main()
{ double xn,xk,dx,dn,xt;

  printf("Уведіть початок інтервалу, що містить єди-
ний корінь:\n");
  scanf("%lf",&xn);
  printf("Уведіть кінець цього інтервалу:\n");
  scanf("%lf",&xk);
  printf("Уведіть точність пошуку кореня:\n");
  scanf("%lf",&dx);
  if(f(xn)*f(xk) >= 0)
    { printf("Умови пошуку кореня не виконують-
ся!\n");
      getch();
```

```
      return;
    }

  dn=xk-xn;
  while(dn > dx)
    { xt=(xk+xn)/2;
      if( f(xn)*f(xt) < 0)
        xk=xt;
      else
        if( f(xn)*f(xt) > 0)
          xn=xt;
        else
          break;
      dn=xk-xn;
    }
 printf("Корінь рівняння: x= %.12f,\n",xt);
 printf("f(x)= %.12f.\n",f(xt));
 getch();
}
```

// Тіло функції:

```
double f(double x)
{
 return 2*sin(5*x)*cos(6*x)+cos(x)-sin(7*x)*cos(4*x);
}
```

Наприклад, для рівняння $2\sin(5x)\cos(6x)+\cos(x)-\sin(7x)\cos(4x)=0$ на інтервалі **[0;0.5]** з точністю **0.0000000001** програма видає:

Корінь рівняння: **x= 0.325635760382,f(x)=** 0.**000000000089**.

Для рівняння $16x-13=0$ на інтервалі **[0;1]** при точності **0.01** (у цьому випадку корінь рівняння співпаде із серединою інтервалу на 4-ому кроці) виходить наступний результат:

Корінь рівняння: **x=0.812500000000, f(x)=** 0.**000000000000**.

12. ВПРАВИ ДЛЯ САМОСТІЙНОЇ РОБОТИ

Розписати по тактах виконання наступних фрагментів програм:

1.
```
int i,s1=0,s2=0;
for(i=1; i < 6; i++)
  { s1+=i*i; s2+=i;}
    s1/=s2;
```

2.
```
int i,im;
float x[7]={3,-1,7,8,9,3,-5};
float xm;
xm=x[0]; im=0;
for(i=1; i < 7; i++)
  if(x[i] > xm)
    { xm=x[i]; im=i;}
```

3.
```
#define N 6
int i;
int a[N]={-3,-2,-1,0,3,4};
for(i=0;i < N; i+=2)
  if(a[i] >= 0)
    a[i]=-3;
  else
    a[i]=3;
```

4.
```
int i=0, a=0, b=1;
while(i <= 5)
  { a+=2; b*=a; i+=2;}
```

5.
```
int i;
int a[7]={3,2,7,8,2,9,2};
for(i=6; i >= 0; i--)
  if(a[i] == 2)
    a[i]=5;
  else
    a[i]=0;
```

```
6.
int i,j;
int a[3][3]={{1,2,3},{4,5,6},{7,8,9}};
for(i=0; i<3; i++)
  for(j=2; j >= 0; j--)
    a[i][j]=-a[j][i];

7.
int I=0,j;
while(i < 1)
  for(j=0;j < i+1;j++)
    i=j%5+1;

8.
int i,j;
int a[5]={4,1,3,1,4};
for(i=0; i < 5; i++)
  { for(j=0; j < a[i]; j++)
      printf("#");
    printf("\n");
  }

9.
int i=0,k;
int a[6]={7,5,3,4,2,8};
while(1)
  { k=a[i+1]%a[i];
    i++;
    if(k == 0)
      goto END;
    else
      printf(" k=%d,",k);
  }
END: printf("\nEND\n");

10.
int k=0,i=0;
int a[6]={2,4,5,0,7,8};
while(k == 0)
  { if(a[i] != 0)
      a[i]=1/a[i];
    else
```

```
      k=1;
    i++;
  }

11.
#define N 6
int i;
char s1[N]="ифраиа";
char s2[N]=".ктмон";
static int s[2*N+1];
for(i=0; i < N; i++)
  { s[2*i]=s1[i];
    s[2*i+1]=s2[N-i-1];
  }
printf("%s\n",s);

12.
int i,j;
int a[3][3];
for(i=0; i < 3; i++)
  for(j=0; j < 3; j++)
    a[i][j]=i*i+j;

13.
int i,j;
int a[3][3];
for(i=2; i >= 0; i--)
  for(j=0; j < 3; j++)
    a[j][i]=(i+1)*j;

14.
int i;
int a[5]={1,4,7,9,10};
int b[5]={3,4,7,10,10};
for(i=0; i < 5; i++)
  { printf("%d=%d ",a[i],b[i]);
    if(a[i]==b[i])
printf("- правильно\n");
else
       printf("- не правильно\n"); }
```

```
15.
int i,j;
for(i=1;i < 5;i++)
  { for(j=1;j < 5;j++)
      printf("%d*%d=%d ",i,j,i*j);
     printf("\n"); }

16.
int i;
char s1[16]="еинавошиярашгошп";
static char s2[17];
char c;
for(i=15;i >=0;i--)
  { c=s1[i];
    if(c == 'ш')
      c='р';
    if(c == 'я')
      c='м';
    s2[15-i]=c;
  }
printf("%s\n",s2);

17.
char c;
char ch[8]="тнедутс";
int i;
for(i=0;i < 3;i++)
  { c=ch[6-i];
    ch[6-i]=ch[i];
    ch[i]=c;
  }
printf("%s",ch);

18.
char ch[8]="меммия";
int i;
for(i=0;i < 6;i++)
  if(ch[i] == 'м')
    ch[i]='с';
 printf("%s",ch);
```

19.
```
char ch[3][4]={"пью","тер","ком"};
static char chn[10];
int i,j;
for(i=0;i < 3;i++)
  for(j=0;j < 3;j++)
     chn[3*i+j]=ch[(3*i*i-7*i+4)/2][j];
printf("%s\n",chn);
```

20.
```
int i, a0=12;
int a[6]={2,7,9,15,12,13};
for(i=0;i < 6 && a[i] != a0; i++);
```

21.
```
  static int a[N][N]={{4,5,1,0,0},
                      {1,1,1,1,1},
                      {1,1,1,1,1},
                      {1,1,1,1,1},
                      {4,1,1,1,0}};
  static int b[N][N]={{2,5,0,0,0},
                      {2,4,6,1,0},
                      {2,4,5,3,0},
                      {2,4,4,5,0},
                      {4,3,7,0,0}};
  int i,j,k;

  for(i=0;i < N;i++)
    { for(j=0;j < N;j++)
        { for(k=0;k < a[i][j];k++)
            printf("*");
          for(k=0;k < b[i][j];k++)
            printf(" ");
        }
      printf("\n");
    }
```

22.
```
int i,j;
  static int a[5]={6,2,6,2,6};
  for(i=0;i < 5;i++)
    { j=0;
```

```
      while(j <= a[i])
        { printf("$");
          j++;
        }
      printf("\n"); }
```

23.

```
int i,j;

  for(i=0;i < 5;i++)
    { for(j=0;j < i;j++)
        printf("#");
      printf("%d\n",i);
    }
```

24.

```
int i,j;

  for(i=0;i < 4;i++)
    { for(j=0;j < 3-i;j++)
        printf(" ");
      for(j=0;j < 2*i+1;j++)
        printf("&");
printf("\n");
    }
```

25.

```
  int i;
  static int a[5]={1,0,3,7,-1};
  static int b[5]={1,0,3,7,-1};

  for(i=0;i < 5;i++)
    { a[i]*=b[4-i];
      b[i]+=a[i];
    }
```

26.

```
  int i;
  static int a[6]={1,3,5,7,11,13};

  for(i=5;i >= 0;i-=2)
    a[i]=-a[5-i];
```

27.

```
  int n,j;
  static int a[6];
  static int p[5]={2,3,5,7,1};

  for(n=8;n <= 13;n++)
    { j=0;
      while(n%p[j] != 0)
        j++;
      if(j == 4)
        a[n-8]=n;
    }
```

28.

```
  int i;
  static int a[6]={1,2,2,-3,4,5};

  for(i=0;i < 5;i++)
    if(a[i]+a[i+1] <= 2*a[i])
      a[i]=0;
    else
      a[i]=1;
```

29.

```
  int i,j;
  static int a[3][3]={{1,2,3},
                      {4,5,6},
                      {7,8,9}};

  for(i=2;i >= 0;i--)
    for(j=0;j < 3;j++)
      a[i][j]=a[(i+2)%3][j];
```

30.

```
  int i,j;
  static int a[3][3];

  for(i=0;i < 3;i++)
    for(j=0;j < 3;j++)
      if((i+j)%2 == 0)
        a[i][j]=1;
```

31.

```
  int i,j;
  static int a[3][3]={{1,2,3},
                      {4,5,6},
                      {7,8,9}};

  for(i=2;i >= 0;i--)
    for(j=0;j < 3;j++)
      if((i*j)%2 == 0)
        a[i][j]*=a[i][j];
```

32.

```
  int i,j;
  static int a[3][3]={{1,2,3},
                      {4,5,6},
                      {7,8,9}};

  for(i=2;i >= 0;i--)
    for(j=0;j < 3;j++)
      if(i > j)
        a[i][j]=0;
```

33.

```
int n, sd=1;
  n=55;
  while(n >= 2*sd)
    sd*=2;
  while(n > 0 || sd > 0)
    if(n >= sd)
      { printf("1");
        n-=sd;
        sd/=2;
      }
    else
      { printf("0");
sd/=2;
      }
  printf("\n");
```

34.

```
int n, sd=1;
  n=401;
```

```
  while(n >= 8*sd)
    sd*=8;
  while(n > 0 || sd > 0)
    if(n >= sd)
      { printf("%d",n/sd);
        n-=sd*(n/sd);
        sd/=8;
      }
    else
      { printf("0");
        sd/=8;
      }
  printf("\n");
```

35.

```
int n, sd=1;
  n=189;
  while(n >= 16*sd)
    sd*=16;
  while(n > 0 || sd > 0)
    if(n >= sd)
      { printf("%X",n/sd);
        n-=sd*(n/sd);
        sd/=16;
      }
    else
      { printf("0");
        sd/=16;
      }
  printf("\n");
```

36.

```
  int i,ab=0;
  static int a[5]={2,3,0,-3,1};
  static int b[5]={1,5,3,-7,2};

  for(i=0;i < 5;i++)
     ab+=a[i]*b[4-i];
```

37.

```
  int i;
  static int a[6]={1,3,5,7,9,0};
```

```
  static int b[6]={1,5,3,9,7,0};

  for(i=0;i < 6;i++)
    { if(a[i] == b[i])
        printf("%d = %d\n",a[i],b[i]);
      if(a[i] > b[i])
        printf("%d > %d\n",a[i],b[i]);
      if(a[i] < b[i])
        printf("%d < %d\n",a[i],b[i]);
}

38.
  int f1=1,f2=1,f3;

  while(f2 <= 100)
    { f3=f1;
      f1=f2;
      f2=f1+f3;
      printf("%d ",f2);}
  printf("\n");
39.
  int i,j;

  for(i=0;i < 4;i++)
    { for(j=0;j < 4;j++)
        { if(i == j)
            printf("O ");
          else
            printf("O ");
}
      printf("\n");
    }

40.
  int i,j;

  for(i=0;i < 3;i++)
    { for(j=0;j < 7;j++)
        { if(i%2 == 0 && (j < 2 || j > 4))
            printf(" ");
          else
            printf("¦");
```

```
        }
      printf("\n");
}
```

41.

```
  int i,b,c;
  static int a[10]={7,3,1,-2,5,10,7,-9,2,4};

  b=c=a[0];
  for(i=1;i < 10;i++)
    { if( a[i] > b)
        b=a[i];
      if( a[i] < c)
        c=a[i];
}
```

42.

```
int n, osnss,zn;

  n=55;
  osnss=2;
  while(n > 0)
    { zn=n%osnss;
      n/=osnss;
      printf("%d",zn);
}
  printf("\n");
```

43.

```
  int n, osnss,zn;

  n=401;
  osnss=8;
  while(n > 0)
    { zn=n%osnss;
      n/=osnss;
      printf("%d",zn);
}
  printf("\n");
```

```
44.
  int n, osnss,zn;

  n=189;
  osnss=16;
  while(n > 0)
    { zn=n%osnss;
      n/=osnss;
      printf("%X",zn);
}
  printf("\n");

45.
char c;
  char ch[14]="анешербчбдбЗ";
  int i;

  for(i=0;i < 6;i++)
    { c=ch[12-i];
      ch[12-i]=ch[i];
      ch[i]=c;
      if(ch[i] == 'б')
       ch[i]='a';
    }
  printf("%s\n",ch);

46.
  char c1[6]="Плчлс";
  char c2[6]="оуиоь";
static char c3[11];
  int i;

  for(i=0;i < 10;i++)
    if(i%2 == 0)
      c3[i]=c1[i/2];
    else
      c3[i]=c2[(i-1)/2];
  printf("%s\n",c3);
```

```
47.
  char c1[8]="студутс";
  char c2[8]="тнедент";
int i,j1,j2;

  for(i=0;i < 7;i++)
    { for(j1=0;j1 < 3-abs(i-3);j1++)
        printf(" ");
      printf("%c",c1[i]);
      if(i != 3)
        { for(j1=0;j1 < abs(2*i-6)-1;j1++)
            printf(" ");
          printf("%c\n",c2[i]);
        }
      else
        printf("\n");
    }

48.
  static char c1[12]="аргентинам";
  static char c2[20];
  int i;

  for(i=0;i < 11;i++)
     c2[i]=c1[i];
  for(i=11;i < 15;i++)
     c2[i]=c1[19-i];
  c2[15]=' ';
  for(i=16;i < 21;i++)
     c2[i]=c1[20-i];

  printf("%s\n",c2);

49.
  #define N 4
  char  a[N][N]={"усту","ессд","сяие","вотн"};

  static char b[N*N+1];
  char i,j=0,k;

  for(k=0;k < N/2;k++)
    { for(i=k;i < N-k;i++)
```

```
        { b[j]=a[k][i];
          j++;
        }
      for(i=k+1;i < N-k-1;i++)
        { b[j]=a[i][N-k-1];
          j++;
        }
      for(i=k;i < N-k;i++)
        { b[j]=a[N-k-1][N-i-1];
          j++;
        }
      for(i=k+1;i < N-k-1;i++)
        { b[j]=a[N-i-1][k];
          j++;
        }
   }
  printf("%s\n",b);
```

50.

```
  static char a[3][3]={{3,1,4},{1,5,9},{2,6,5}};
  static char b[3]={2,0,1};
  char i,k;

  for(k=0;k < 3;k++)
    for(i=k;i < 3;i++)
      a[k][i]=a[b[i]][b[k]];
```

Приклад виконання завдання (№ 5):

1-ий крок: **i=6, 6 >= 0** - вірно, **a[6]=2 == 2** - вірно, **a[6]=5**, **a={3,2,7,8,2,9,5}, i=6-1=5**.

2-ий крок: **5 >= 0** - вірно, **a[5]=9 == 2** - не вірно, **a[5]=0**, **a={3,2,7,8,2,0,5}, i=5-1=4**.

3-ій крок: **4 >= 0** - вірно, **a[4]=2 == 2** - вірно, **a[4]=5**, **a={3,2,7,8,5,0,5}, i=4-1=3**.

4-ий крок: **3 >= 0** - вірно, **a[3]=8 == 2** - не вірно, **a[3]=0**, **a={3,2,7,0,5,0,5}, i=3-1=2**.

5-ий крок: **2 >= 0** - вірно, **a[2]=7 == 2** - не вірно, **a[2]=0**, **a={3,2,0,0,5,0,5}, i=2-1=1**.

6-ий крок: **`1 >= 0`** - вірно, **`a[1]=2 == 2`** - вірно, **`a[1]=5`**, **`a={3,5,0,0,5,0,5}, i=1-1=0.`**

7-ий крок: **`0 >= 0`** - вірно, **`a[0]=3 == 2`** - не вірно, **`a[0]=0`**, **`a={0,5,0,0,5,0,5}, i=0-1=-1`**.

8-ий крок: **`-1 >= 0`** - не вірно, кінець циклу.

13. ЗАДАЧІ ДЛЯ САМОСТІЙНОГО СКЛАДАННЯ ПРОГРАМ

13.1. Задачі І-го рівня складності

1.1. Ділянка колгоспного поля має форму чотирикутника, у якого дві сторони довжиною ***a, з*** рівнобіжні, а третя сторона довжиною ***b*** перпендикулярна до них. Щоб огородити ділянку забором, треба визначити периметр ділянки $L = a + b + c + \sqrt{b^2 + (a-c)^2}$. Скласти програму для обчислення периметра.

Контрольний варіант вхідних даних:
a = 462 м, b = 195 м, c = 287 м.

1.2. Два села колгоспу ***A*** *і* ***B*** знаходяться відповідно на відстанях ***a, b***від газової магістралі і віддалені одне від другого на відстань ***c.*** Для їхньої газифікації треба побудувати газорозподільний пункт, з'єднавши його газопроводами із селами. Відстані від сіл до газорозподільного пункту, при яких довжина всього газопроводу найменша, виражаються формулами:

$$l_1 = \frac{a}{a+b}\sqrt{(a+b)^2 + c^2},\ l_2 = \frac{b}{a+b}\sqrt{(a+b)^2 + c^2}.$$

Скласти алгоритм для обчислення відстаней l_1, l_2 і сумарної довжини газопроводу $l = l_1 + l_2.$.

Контрольний варіант вхідних даних: ***a = 3.62 км, b = 2.47 км, c = 4.39 км.***

1.3. Ставок має форму кола. Частина його, обмежена хордою, заболотилась і заросла очеретом. Довжина берегової лінії (частина кола) не заболоченої частини ставка L, довжина берегової лінії заболоченої частини ставка l. Скласти алгоритм для обчислення повної площі ставка $S = \frac{1}{4\pi}(L+l)^2$, площі не заболоченої частини ставка $\sigma = \frac{1}{4\pi}L(L+l) + \frac{1}{8\pi^2}(L+l)^2 \sin\left(\frac{2\pi l}{L+l}\right)$ і показника заболоченості ставка у відсотках $P = 100(1 - \sigma / S).$.

Контрольний варіант вхідних даних: $L = 114$м, $l = 38$м.

1.4. Скласти програму обчислення площі трикутника по трьох сторонах за формулою Герона.

1.5. Скласти програму обчислення площі трикутника по двох сторонах і кутові між ними.

1.6. Скласти програму обчислення радіуса вписаного до трикутника кола й обчислення радіуса описаної навколо трикутника кола.

1.7. Судина масою m_0 і місткістю V_0 заповнена землею. При зважуванні судини до і після висушування землі отримано значення маси m_B і m_C. Знаючи густину $\rho_Г$ речовини ґрунту і щільність ρ води, скласти програму для визначення у відсотках відносних об'ємів води $c_B = \dfrac{100(m_B - m_C)}{\rho V_0}$ і повітряних пір $c_П = 100\left[1 - \dfrac{m_C - m_0}{\rho_Г V_0}\right] - c_B$ у ґрунті.

Контрольний варіант вхідних даних:

$V_0 = 1000см^3, m_0 = 250г, m_B = 1770г, m_C = 1367г, \rho = 1г/см^3, \rho_Г = 2.6г/см^3.$

1.8. Скласти програму пошуку в масиві заданого числа.

1.9. Скласти програму обчислення визначника 3-го порядку.

1.10. Скласти програму обчислення відстані між двома заданими точками у просторі.

1.11. Колгоспники прийняли зобов'язання зібрати пшеницю з площі ***S [га]*** за ***N*** днів. Фактична продуктивність праці $P_Ф$ протягом перших днів збирання виявилася вище планованої $P_П$, що дозволило за ***M*** днів зібрати пшеницю з площі ***Z [га](Z<S)***. У наступні дні через несприятливі погодні умови виникла необхідність прискорити збирання врожаю і завершити її на ***R*** днів раніше запланованого терміну. Скласти програму обчислення запланованої продуктивності праці $P_П$, фактичної продуктивності праці $P_Ф$ протягом перших днів збирання врожаю і продуктивності праці $P_Т$ в непогожі дні, використовуючи рівняння:

$$P_П N = S, P_Ф M = Z, P_Т(N - M - R) = S - Z.$$

Контрольний варіант вхідних даних:

$S = 1500га, Z = 420га, N = 12днів, M = 3дня, R = 2дня.$

1.12. При переробці соняшника одержують по вазі ***$P_М$%*** олії, ***$P_Ж$%*** макухи і ***$P_Л$%*** лушпиння. Приймаючи щільність соняшникової олії рівною ρ, скласти програму для обчислення маси ***M*** соняшника, необхідного для ви-

робництва ***V*** літрів соняшникової олії, а також маси ***G*** макухи, яка утворяться при цьому, за формулами .

$M = 100\rho V / P_{\text{М}}, G = MP_{ж} / 100$

Контрольний варіант вхідних даних:

$P_{\text{М}} = 44\%, P_{ж} = 28\%, \rho = 0.93кг / л, V = 1500л.$

1.13. Ремонтною майстернею було заплановано відремонтувати ***T*** тракторів за ***D*** днів. Фактично за час ремонту в майстерню надійшло ще ***R*** тракторів. Перевиконуючи план ремонтних робіт щодня на ***P%***, майстерня завершила ремонт усіх тракторів, що надійшли, ***на***N днів раніше запланованого терміну. Скласти алгоритм обчислення ***N***, вважаючи, що це число є цілим: $N = \operatorname{int} K, K = D\left(\text{P} - 100\frac{R}{T}\right)/\left(P + 100\right), \left(P \geq 100\frac{R}{T}\right).$

Контрольний варіант вхідних даних:

T=37 тракторів, R=2 трактора, D=125 днів, P=8%.

1.14. Зважування поросяти показало, що його маса за ***N*** днів збільшилася від M_1 до M_2 $[кг]$. Вважаючи щоденний приріст ваги поросяти ***P*** (у %) постійним, скласти алгоритм для обчислення приросту за формулою $\text{P} = 100\left[\left(\text{M}_2 / \text{M}_1\right)^{1/N} - 1\right].$

Контрольний варіант вхідних даних: ***M_1=35 кг, M_2=51 кг, N=27 днів.***

1.15. Горизонтальний перетин курника являє собою прямокутник довжиною l і шириною b. Плаский дах курника нахилено до обрію під кутом α і висунуто за межі стін на відстань c. При підготовці до робіт по утепленню даху виникла необхідність визначити його площу $S = \frac{(b + 2c)(l + 2c)}{\cos\alpha}$. Скласти програму обчислення S.

Контрольний варіант вхідних даних:

$l = 18м, b = 10.2м, c = 0.3м, \alpha = 20^0.$

1.16. Скласти програму відшукання відстані між площиною і точкою в просторі.

1.17. Скласти програму визначення кута між двома прямими на площині.

1.18. Силосна яма глибиною h має форму усіченої правильної чотирикутної піраміди зі стороною квадратної основи на поверхні землі a і кутом нахилу бічної стінки до обрію α. Скласти програму обчислення

об'єму силосної ями $V = h(a^2 - \frac{2ah}{tg\alpha} + \frac{4h}{3tg\alpha})$ і площі облицювального матеріалу, використаного для покриття дна і бічної поверхні $S = a^2 + 4ahtg\frac{\alpha}{2} - 4h^2\frac{tg\frac{\alpha}{2}}{tg\alpha}$. Додатково програма повинна забезпечувати перевірку коректності вхідних даних.

Контрольний варіант вхідних даних: $h = 3.2$ м, $a = 4.5$ м, $\alpha = 75^o$.

1.19. При цілком відкритому водопровідному крані вода, що випливає з нього, наповнює бак ємністю ***G*** за час ***T***. Скласти програму для визначення тиску води у водопровідній трубі $P = P_a + \frac{\rho G^2}{2kgT^2}(\frac{1}{\sigma^2} - \frac{1}{S^2})$, де $\boldsymbol{P}_a$- атмосферний тиск, ρ - щільність води, ***S*** - площа поперечного перетину водопровідної труби, σ- площа поперечного перетину отвору клапана повністю відкритого крана, ***g*** - прискорення сили ваги, ***k*** - коефіцієнт перерахування тиску в атмосфери.

Контрольний варіант вхідних даних:

$P_a = 1атм, \rho = 1г/см^3, \mathrm{G} = 2\cdot 10^4 см^3, \sigma = 0.2см^2, T = 120с, S = 3.2см^2, k = 1000.$

1.20. Добовий раціон поросяти повинний містити $A[кг]$ кормових одиниць і $B[кг]$ протеїну. У кілограмі першого виду кормів (наприклад, макухи) міститься $G_K[кг]$ кормових одиниць і $G_П[кг]$ протеїну; у кілограмі другого виду кормів (наприклад, кукурудзи) міститься $K_K[кг]$ кормових одиниць і $K_П[кг]$ протеїну. Скласти програму розрахунку необхідних відносних вмістів першого

$P_1 = [(AK_П - BK_K)/(AK_П - BK_K - AG_П + BG_K)]\cdot 100\%$

і другого $P_2 = 100\% - P_1$ видів кормів у раціоні поросят. Програма повинна забезпечувати видачу повідомлення про неможливість одержати з даних компонентів кормову суміш із необхідними характеристиками, якщо в результаті обчислень виявляється $P_1 < 0$ або $P_1 > 100\%$.

Контрольний варіант вхідних даних:

$A = 2.7кг, B = 0.26кг, G_K = 1кг, G_П = 0.4кг, K_K = 1.25кг, K_П = 0.08кг.$

1.21. Ділянка колгоспного поля має форму чотирикутника ***ABCD*** зі сторонами ***AD = a, AB = b, CD = c*** і кутами α при вершині ***A*** и β при ве-

ршині ***D***. Максимальний лінійний розмір поля представляє найбільше число із шести величин: довжин чотирьох сторін і двох діагоналей. Скласти програму обчислення площі ділянки $S = \frac{ab\sin\alpha + ac\sin\beta - bc\sin(\alpha+\beta)}{2}$ і її максимального лінійного розміру $L = \max(a,b,c,L_1,L_2,L_3)$, де

$L_1 = \sqrt{a^2 + b^2 + c^2 - 2ab\cos\alpha - 2ac\cos\beta + 2bc\cos(\alpha+\beta)}$ - довжина сторони ВС; $L_2 = \sqrt{a^2 + b^2 - 2ab\cos\alpha}$, $L_3 = \sqrt{a^2 + c^2 - 2ac\cos\beta}$ - довжини діагоналей.

Контрольний варіант вхідних даних:

$a = 552м, b = 735м, c = 914м, \alpha = 113^0, \beta = 147^0.$

1.22. Потрібно укрити від непогоди скирту соломи висотою ***h***. Основа скирти являє собою прямокутник зі сторонами ***l, b,*** бічні грані утворюють з основою кут ***α***, і верхня грань рівнобіжна основі. Скласти програму обчислення площі поверхні скирти $S = bl + 2h(b+l)tg\frac{\alpha}{2} - \frac{4h^2 tg\frac{\alpha}{2}}{tg\alpha}$.

Контрольний варіант вхідних даних:

$l = 26.1м, b = 7.7м, h = 5.1м, \alpha = 72^0.$

1.23. Горизонтальний перетин скирти соломи, що має форму усіченої чотирикутної піраміди, на довільній висоті x $(0 \le x \le h)$ являє собою прямокутник зі сторонами $l - \frac{2x}{tg\alpha}$ і $b - \frac{2x}{tg\alpha}$, де ***α*** – кут нахилу бічних граней до підстави. Скласти програму обчислення маси *M* скирти, якщо щільність соломи ρ.

Контрольний варіант вхідних даних:

$l = 23.7м, b = 8.36м, h = 5.7м, \alpha = 75^0, \rho = 600кг/м^3.$

1.24. Добова норма годівлі однієї корови складає ***A [кг]***, одного коня - ***У [кг]*** сіна. Скласти програму обчислення найбільшого числа корів ***K***, яких можна прокормити протягом ***N*** днів, при тому, що маємо сіно масою ***P [кг]***, і якщо при цьому прийдеться одночасно утримувати ***L*** коней ***(P > BLN)***, а також маси сіна ***R [кг]***, яке при цьому залишається.

Контрольний варіант вхідних даних:

N=120 днів, L=5, P=55000 кг, A=6 кг, У=9 кг.

1.25. Маємо мультимедіа-плеєр з функцією безпровідної мережі (WiFi). Реальна швидкість передачі інформації по безпровідній мережі 27 Мбіт/сек. Розмір середнього фільму HD-якості тривалістю 1,5 години складає 4 Gb. Чи вистачить пропускної здатності мережі для безперервного перегляду відеофільму в режимі реального часу? Якого максимального об'єму фільм (при тій же тривалості 1,5 год.) можна переглядати без проблем за допомогою даної безпровідної мережі?

1.26. В бочку налито **M** літрів **100%** спирту. Кухлем, що має об'єм **N** літрів, забирають спирт і наливають воду. Суміш ретельно перемішують і операцію повторюють. Розрахувати обсяг чистого спирту в бочці, води і процентний вміст спирту в суміші за кількість операцій від **1** до **K**.

Привести результати роботи програми для:
M=120л, N=3л, K=100.

1.27.Взаємний вплив двох конкуруючих видів тварин у Африці – тигрів і леопардів – на розмір x_n, y_n їх популяцій у n-му році можна описати системою:

$$\begin{cases} x_{n+1} = 2x_n - y_n \\ y_{n+1} = -x_n + 2y_n \end{cases}$$

Хай ***x0=a, y0=b (a≠b)***, де a і b – задані числа. Знайти чисельність обох видів за всі роки, що передують повному вимиранню одного з двох видів.

Навести результати роботи програми для:

а) a=200650; b=200630;

б) a=2500820; b=2500830.

13.2. Задачі середньої складності

2.1. Скласти програму, яка обчислює за алгоритмом Евкліда найбільший загальний дільник (НЗД) двох чисел.

2.2. Скласти програму рішення лінійної системи II-го порядку.

2.3. За даними дослідної станції, для одержання високого врожаю жита на відведених для неї колгоспом площах до ґрунту щорічно на кожен гектар варто вносити у вигляді органічних добрив ***N [кг]*** азоту і ***K [кг]*** калію. Ваговий вміст цих елементів у ***1 т*** свіжого гною складає A_c ***і*** B_c***[кг],*** а

в ***1 т*** перепрілого гною складає A_n ***і*** B_n***[кг]*** відповідно. Скласти програму обчислення необхідних мас ***X*** свіжого і ***Y*** перепрілого гною, які необхідно щорічно вносити до ґрунту на гектар підготовлених для посівів жита площ для виконання рекомендацій дослідної станції.

Контрольний варіант вхідних даних: ***N=3.2 кг, K=3.9 кг,***

Вид добрива	Вміст елементів у 1 т, кг	
	Азот	Калій
Свіжий гній	5	6
Перепрілий гній	6	7.5

2.4. Скласти програму відшукання мінімальної і максимальної координати ***n***-вимірного вектора.

2.5. Скласти програму, яка використовуючи формулу ***n*m= НЗД(n,m)*НЗК(m,n)*** знаходить найменше загальне кратне двох чисел.

2.6. Внутрішнє приміщення корівника має форму прямокутного паралелепіпеда довжиною ***l***, шириною ***b*** і висотою ***h***. Для очищення повітря в корівнику встановили витяжний вентилятор із коефіцієнтом корисної дії ***p%***, що повинний перекачувати за час ***T*** через круглий вентиляційний канал діаметром ***d*** об'єм повітря, який дорівнює об'ємові приміщення корівника. Складіть програму обчислення продуктивності вентилятора $Q = \frac{lbh}{T}$, швидкості повітря у вентиляційному каналі $\nu = \frac{4Q}{\pi d^2}$ і потужності електричної енергії, споживаної вентилятором, $M = \frac{12.5\pi\rho d^2 v^3}{pG}$, де $\rho = 1.3 кг/м^3$ – щільність повітря, $G = 1 Н \cdot м / Дж$ – механічний еквівалент електричної енергії. Алгоритм повинний забезпечувати видачу результатів у вигляді таблиці з пояснювальним текстом для значень ***T*** від T_{min} до T_{max} з інтервалом ΔT.

Контрольний варіант вхідних даних:

$l = 182 м, b = 16 м, h = 3.8 м, p = 40\%, d = 0.4 м,$

$T_{\min} = 3600 с, T_{\max} = 10800 с, \Delta T = 1800 с.$

2.7. Фермер запланував підвищити щорічні надої молока з A_0у ***2003*** році до A_n у ***2003+n*** року. Необхідно розподілити необхідні щорічні надої молока $A_1, A_2, \ldots, A_n$ таким чином, щоб щорічний приріст виробництва мо-

лока ***P*** (у %), щодо рівня виробництва попереднього року, залишався постійним. Скласти програму обчислення значень ***P*** і ***A_1 , A_2 ,…A_n*** відповідно до співвідношень $A_i = A_0(1+\frac{P}{100})^i$, де ***i=1,2,…n.***

Контрольний варіант вхідних даних: ***A_0=24000л, A_n=28000л, n=5.***

2.8. Скласти програму, яка друкує спільні букви в двох заданих словах.

2.9. Скласти програму обчислення середнього і дисперсії вибірки.

2.10. Скласти програму обчислення коефіцієнта кореляції між двома вибірками.

2.11. Скласти програму визначення по даті дня тижня.

2.12. Скласти програму визначення числа днів між двома датами.

2.13. Скласти програму ранжування координат ***n-вимірного*** вектора в порядку зростання й убування.

2.14. Скласти програму друку простих чисел у заданому діапазоні.

2.15. Два села ***A*** і ***B*** розташовані на березі річки на відстані ***L*** одне від другого, третє село ***Ci*** знаходитися на тому ж боці річки і віддалене від сіл ***A, B*** на відстані відповідно ***a*** і ***b*** (русло річки вважається прямолінійним). Якщо пристань ***П*** побудувати на відстані ***x*** від села ***A***, то відстані від пристані до сіл ***A, B, C*** будуть виражатися відповідно формулами

$$S1 = x, \mathrm{S2} = \mathrm{L} - \mathrm{x}, \mathrm{S3} = \sqrt{\mathrm{a}^2 + \mathrm{x}^2 - \frac{\mathrm{x}}{\mathrm{L}}(\mathrm{a}^2 - \mathrm{b}^2 + L^2)}.$$

Жителям сіл необхідно вирішити, в якому місці побудувати пристань, щоб сумарна відстань від пристані до сіл ***S=S1+S2+S3*** була найменшою. Для допомоги жителям сіл скласти програму:

а) яка друкує таблицю залежності сумарної відстані ***S*** від значення ***x*** із кроком ***dx*** від ***0*** до ***L ;***

б) яка друкує оптимальні відстані ***$x_{опт}$, $S_{опт}$***.

Контрольний варіант вхідних даних: ***L=60км, a=45км, b=35км, dx=5км.***

2.16. Первісна вартість устаткування міжколгоспної ремонтної майстерні складає R_0 грн. Щорічно на суму ***D*** грн. закуповували нове обладнання. Щорічна амортизація (тобто зменшення вартості) наявного устаткування складає ***P %*** від його вартості. Скласти програму для обчислення

вартості устаткування ремонтної майстерні R_N через ***N*** років після введення її до експлуатації відповідно до співвідношення $R_N = R_{N-1}\left(1-\frac{P}{100}\right)$, де $\boldsymbol{R}_N$ - вартість устаткування у ***N-му*** році, ***N≥1***.

Контрольний варіант вхідних даних:

$R_0 = 45000грн, D = 4000грн, P = 7.7\%, N = 10.$

2.17. Ділянку поля, що має форму чотирикутника ***ABC*** (рис.13.1), обмежено лініями $AD = a, AB = b, BC = c, d \le c$ і ***CD***, причому ***AD*** рівнобіжна до ***BP***, а ***AB*** перпендикулярна до ***AD***. Поле потрібно розділити на дві частини відрізком прямої лінії ***EF***, рівнобіжним межі ***AB*** і розташованим від неї на відстані ***d*** так, щоб відношення площі S_1 прямокутної ділянки ***ABEF*** до площі S_2 ділянки ***EFDC*** дорівнювало ***k***. Скласти програму обчислення відстані ***d***, а також площ S_1, S_2 і $S = S_1 + S_2$.

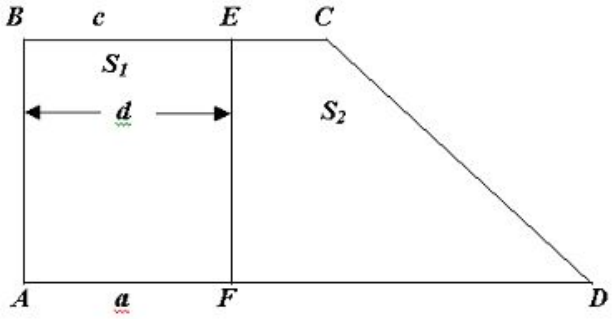


Рис. 13.1

Контрольний варіант вхідних даних: ***a=1643м, b=784м, з=1279м, k=3.***

2.18. Поперечний переріз закритого каналу зрошувальної системи, має площу $S = a(b+h) - h^2 / tg\alpha$ і периметр $P = 2(a + b + htg\frac{\alpha}{2})$. Скласти програму визначення висоти покриття h при заданих значеннях a, b, α, для якої відношення $Z = S / P$ має найбільше значення $(0 \le h < \frac{a}{2} tg\alpha)$.

Контрольний варіант вхідних даних: $a = 3.5м, b = 1.5м, \alpha = 40^0$.

2.19. Потрібно побудувати зрошувальний канал із поперечним перерізом площею ***S*** у формі рівнобічної трапеції і з кутом α нахилу бічних стінок до обрію таким чином, щоб на його облицювання було використано найменшу кількість матеріалу. Площа поверхні одиниці довжини каналу

σ виражається через глибину каналу h формулою $\sigma = \frac{S}{h} - \frac{h}{tg\alpha} + \frac{2h}{\sin\alpha}$. При цьому ширина каналу $b = \frac{S}{h} + \frac{h}{tg\alpha}$. Скласти програму обчислення оптимальних значень h і b.

Контрольний варіант вхідних даних: $S = 1м^2, \alpha = 50^0$.

2.20. Потрібно виготовити шухляду, яка закривається, зображену на рис. 13.2. Площа поверхні шухляди ***S*** виражається через її об'єм ***V*** співвідношенням $S = 2V\left(\frac{1}{a} + \frac{1}{b}\right) + ab\frac{1+\cos\alpha}{\cos\alpha}$, при цьому висота ***H*** визначається формулою $H = \frac{V}{ab} + \frac{1}{2}btg\alpha$. Скласти програму для обчислення лінійних розмірів шухляди a, b, H і площі її поверхні ***S***, за якими на виготовлення шухляди із заданим об'ємом ***V*** потрібно найменшу кількість матеріалу (площа поверхні мінімальна).

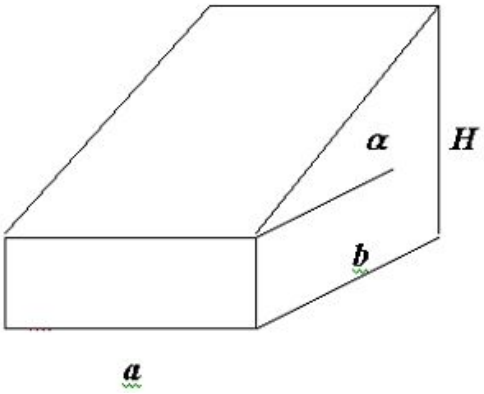


Рис. 13.2

Контрольний варіант вхідних даних: $V = 0.1м^3, \alpha = 35^0$.

2.21. При огляді контрольної ділянки лісу лісник підрахував чисельність дерев на цій ділянці: сосон n_1, єлей n_2, беріз n_3 і осик n_4, причому з них здоровими є відповідно: сосон m_1, ялин m_2, беріз m_3 і осик m_4. Інші дерева хворі або мають ушкодження. Скласти програму обчислення наступних показників стану лісу: числа дерев на контрольній ділянці $n = \sum_{i=1}^{4} n_i$, числа здорових дерев $m = \sum_{i=1}^{4} m_i$, чисельності різних видів дерев у процентному відношенні: $\mathrm{P}_i = 100\frac{n_i}{n}$, чисельності хворих дерев у процент-

ному відношенні: $q = 100\left(1 - \frac{m}{n}\right)$ для ділянки лісу в цілому і $q_i = 100\left(1 - \frac{m_i}{n_i}\right)$ для кожного з видів дерев $(i = 1,...,4)$.

Контрольний варіант вхідних даних:

Види дерев	Сосни	Ялини	Берези	Осики
Загальна чисельність	64	95	56	47
Здорових дерев	58	89	48	44

2.22. Електроустаткування корівника споживає ***K* [квт]** електроенергії від лінії електропередачі через підстанцію з вихідною напругою ***V* [В]**, віддалену від корівника на відстань ***L* [м]**. Скласти програму обчислення діаметра $d = 10^6\sqrt{\frac{40\rho LK}{\pi V^2 p(100-p)}}$ (у мм) провідника лінії електропередачі з питомим опором ρ, при якому в лінії за рахунок активного опору проводів втрачається ***p%*** переданої електроенергії. Програма повинна забезпечити видачу результатів у вигляді таблиці з пояснювальним текстом для значень ***p*** від p_{min} до p_{max} з інтервалом ***dp***.

Контрольний варіант вхідних даних:

$\rho = 2.7 \cdot 10^{-8}, L = 600м, K = 15кВт, V = 220В, p_{\min} = 0.5\%, p_{\max} = 5\%, dp = 0.5\%.$

2.23. Скласти програму рішення рівняння $ax^2+bx+c=0$. Програма повинна видавати правильну відповідь при будь-яких значеннях коефіцієнтів ***a, b*** і ***c***.

2.24. Автомобіль витрачає Q[л] бензину на ***100 км*** шляху, що можна розрахувати за формулою $Q = (av - b + c/v)e^{kv}$, де v - швидкість автомобіля, a, b, c, k - коефіцієнти, що залежать від ходових властивостей автомобіля. Скласти програму:

а) обчислення витрати бензину автомобілем Q у залежності від швидкості v з заданим кроком Δv у заданому інтервалі $[v_n, v_k]$.

б) обчислення найбільш економічної швидкості автомобіля $v_э$, витрати бензину $Q_э$, яка відповідає цій швидкості, найменшої і найбільшої швидкості, за яких витрата бензину перевищує $Q_э$ не більше ніж на p%. Точність обчислення швидкостей ***0.1 км.***

Контрольний варіант вхідних даних:

$a = 0.21$ ***л·ч/км,*** $b = 18$ ***л/км,*** $c = 760$ ***л/ч,*** $k = 0.005$ ***ч/км,*** $\Delta v = 5$ ***км/год,***

$v_n = 10$ ***км/год,*** $v_k = 150$ ***км/год,*** $p =$ ***20%.***

2.25. У деяких видах спортивних змагань виступ кожного спортсмена незалежно оцінюється декількома суддями, потім із усієї сукупності оцінок видаляються найбільш висока і найбільш низька, а для оцінок, що залишилися, обчислюється середнє арифметичне, котре і йде до заліку спортсменові. Якщо найбільш високу оцінку виставило декілька суддів, то із сукупності оцінок видаляють тільки одну таку оцінку, аналогічно роблять і з найбільш низькими оцінками.

Є натуральне число ***n,*** дійсні позитивні числа a_1, a_2, ..., a_n. Вважаючи, що числа a_1, a_2, ..., a_n — це оцінки, виставлені суддями одному з учасників змагань, визначити оцінку, що піде до заліку цьому спортсменові.

Контрольні варіанти вхідних даних:

n=10, a=(3,4,3,5,5,4,2,3,4,5);

n=5, a=(3,4,4,5,6);

n=11, a=(5,5,5,5,5,5,5,5,5,5,5).

2.26. Табулювання функції

$$F(x) = \begin{cases} -12x, & \text{при } x \le -3 \\ e^{\sqrt{x+3}}, & \text{при } -3 < x < 4 \\ \ln x - \log_3 x, & \text{в інш. випадках} \end{cases}$$

на заданому інтервалі ***[a,b]*** із заданим кроком Δx.

Контрольний варіант вхідних даних: ***a = -5, b = 8,*** $\Delta x = 0.3$.

2.27. У вертикально розташованій циліндричній бочці з діаметром основи d, заповненої до висоти H_0 бензином, утворилася теча із площею поперечного переріза σ, розташована на відстані h від основи $(h < H_0)$. Течу помітили через деякий час T_1. Відомо, що швидкість v витікання бензину через отвір визначається формулою $v = \sqrt{2g(H - h)}$, де H - висота рівня бензину (H зменшується протягом часу t), g -постійне прискорення сили тяжіння. Відомо також, що об'єм бензину Q, що витікає через отвір

за одиницю часу, виражається формулою $Q = v\sigma$. Скільки відсотків p бензину виявилося втраченим до моменту виявлення течі?

Контрольний варіант вхідних даних:

$d = 0.6м, \mathrm{H}_0 = 0.8м, S = 2 \cdot 10^{-6} м^2, h = 0.3м, T_1 = 3600с.$

2.28. Знайти на заданому відрізку ***[m,n]*** усі досконалі числа (число називається досконалим, якщо сума його дільників, за винятком самого числа, дорівнює йому самому, наприклад: ***6 = 1+2+3***).

Контрольний варіант вхідних даних: ***m = 8, n=10000***.

2.29. На сільгосппідприємстві побудували цех по виготовленню кормів. Протягом першого місяця роботи цеху було введено до дії частину устаткування, проектна продуктивність якого складає $Q\%$ від проектної продуктивності всього цеху. Інше устаткування буде введено до дії через N місяців. Протягом першого місяця роботи продуктивність нового обладнання складе $R\%$ проектної, а протягом кожного наступного буде збільшуватися на $S\%$ від продуктивності попереднього місяця, поки не досягне проектної. Скласти алгоритм розрахунку продуктивності цеху $P\%$ від проектної по місяцях M до досягнення цехом повної віддачі:

$$P = \begin{cases} P_2 & \text{при } M \le N, \\ P_2 & \text{при } M \ge N+1, P_3 < 100, \\ 100 & \text{при } M \ge N+1, P_3 \ge 100. \end{cases}$$

де

$$P_2 = \begin{cases} P_1 & \text{при } P_1 \le Q, \\ Q & \text{при } P_1 > Q. \end{cases}$$

$$P_3 = P_2 + \frac{(100-Q)R}{100}\left(1+\frac{S}{100}\right)^{M-N-1};$$

$$P_1 = \frac{QR}{100}\left(1+\frac{S}{100}\right)^{M-1}$$

Контрольний варіант вхідних даних: $Q = 40\%, R = 20\%, S = 30\%, N = 6.$

2.30. Після чергового прополювання поля сумарні біомаси корисних рослин і бур'янів складають відповідно ***M_0*** i ***$S_0 (S_0 << M_0)$***. Надалі біомаси

M_i корисних рослин і S_i бур'янів у довільний ***i-й*** момент часу $t = i\Delta t$ у період інтенсивного зростання рослин визначаються рівняннями:

$$\frac{M_{i+1} - M_i}{\Delta t} = M_i(k_1 - \frac{k_2}{\sigma} S_i), \quad \frac{S_{i+1} - S_i}{\Delta t} = S_i(k_3 - \frac{k_4}{\sigma} M_i),$$

де $i = 0, 1, 2, ..., n$; Δt— малий інтервал часу; σ — площа поля; k_1, k_2, k_3, k_4 — коефіцієнти, які характеризують швидкість росту рослин. Скласти програму:

а) обчислення біомаси M_i корисних рослин, біомаси S_iбур'янів і їхнього відношення $Z_i = S_i/M_i$для кожного моменту часу в заданому інтервалі.

б) обчислення часу $T = m\Delta t$ після прополювання, після закінчення якого біомаса корисних рослин при відсутності повторного прополювання почне убувати через заростання поля бур'янами.

Контрольний варіант вхідних даних:

$M_0 = 7.4, S_0 = 0.015, \sigma = 8, k_1 = 0.035, k_2 = 0.037, k_3 = 0.167, k_4 = 0.013, \Delta t = 1.$

2.31.Один з художників вважає, що найбільш доцільними розмірами полотен для його картин є такі, за яких площа полотна чисельно дорівнює його периметру. Знайдіть всі улюблені **цілочисельні** розміри полотен художника у ***дм***, за умови, що довжина одної сторони полотна не повинна перевищувати ***25 дм***.

13.3. Задачі підвищеної складності

3.1. Голова колгоспу виїхав до районного центру, розташованого на відстані ***S***, на автомобілі, що рухається зі швидкістю ***Vn***. Через час T_0 слідом за ним виїхав агроном, рухаючись на мотоциклі зі швидкістю ***Va (Va > Vn)***. Відомо, що голова по дорозі затримався на чаювання протягом часу ***Тч*** у сусідньому селі, відстань до якого ***S1 (S1 < S)***. Через який час, і на якій відстані від свого села агроном наздожене голову?

Контрольні варіанти вхідних даних:

1) S=60км, S1=20км, Vn=50км/год, Va=70км/год, T_0=6хв, Тч=18хв.

2) S=60км, S1=10км, Vn=50км/год, Va=60км/год, T_0=12хв, Тч=12хв.

3) S=60км, S1=10км, Vn=50км/год, Va=60км/год, T_0=12хв, Тч=6хв.

4) S=40км, S1=10км, Vn=55км/год, Va=60км/год, T_0=12хв, Тч=6хв.

3.2. Скласти програму обчислення зворотної матриці 3-го порядку.

3.3. Щоденний приріст P_n маси M_n поросяти віком ***n*** днів визначається за приблизною формулою $P_n = \lambda \cdot (M^* - M_{n-1})$, а щоденна витрата K_n кормів для поросяти у цьому віці складає $K_n = \mu \cdot M_n$. Тут M^* - маса дорослої тварини, λ, μ - коефіцієнти, які характеризують швидкість росту і норму споживання кормів, $M_n = M_0 + \sum_{j=1}^{n} P_j$, M_0 - маса поросяти при народженні. Прибуток від здавання поросяти на м'ясозаготівельний пункт віком ***n*** днів визначається за виразом $D_n = C_m \cdot M_n - C_k \cdot \sum_{i=1}^{n} K_i - n \cdot C_y - C_0$,

де C_m - вартість одиниці маси живої ваги поросяти,

C_k - вартість одиниці маси кормів,

C_y - витрати на догляд і утримання приміщення,

C_0 - початкові витрати.

Вважаючи параметри $M_0, M^*, \lambda, \mu, C_m, C_k, C_y, C_0$ відомими і незалежними від ***n***, скласти програму, яка друкує прибуток від здавання поросяти на м'ясозаготівельний пункт віком ***n*** днів для кожного ***n*** в інтервалі ***1*** до ***n_{max}***. Причому цей інтервал повинний містити значення ***n***, за яким досягається максимальний прибуток.

Контрольний варіант вхідних даних:

$M_0 = 5кг, M^* = 200кг, \lambda = 0.003сут^{-1}, \mu = 0.08сут^{-1}, C_m = 1.4грн/кг,$

$C_k = 0.05грн/кг, C_y = 0.2грн/сут, C_0 = 11грн.$

3.4. Скласти програму розв’язання лінійної системи 3-го порядку.

3.5. Силосна вежа має форму, яку показано на рис. 13.3. Нижня частина вежі – циліндр із діаметром основи d_H і висотою h_H, верхня частина – усічений конус із діаметром нижньої основи d_H, діаметром верхньої основи d_B і висотою h_B. Використаний об’єм V силосної вежі при заповненні до висоти h визначається за формулами:

$$V=\begin{cases}\dfrac{1}{4}\pi d_H^2 h, \text{ при } 0\le h\le h_H \\ \dfrac{1}{4}\pi d_H^2 h_H+\dfrac{\pi d_H^3 h_B}{12(d_H-d_B)}\left\{1-\left[1-\dfrac{h-h_H}{h_B d_H}(d_H-d_B)\right]^3\right\}, \text{ при } h_H<h\le h_H+h_B\end{cases}$$

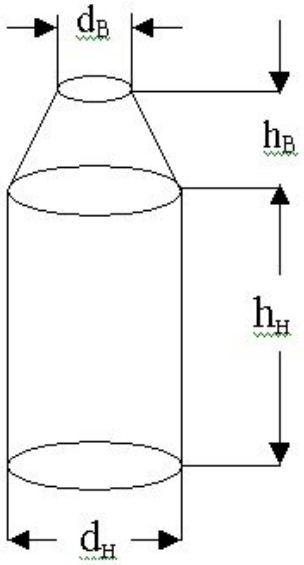


Рис. 13.3

Скласти програму обчислення висоти заповнення силосної вежі при корисному використанні $p\%$ її об'єму.

Контрольні варіанти вхідних даних:

$d_H=7.08$ м, $d_B=3.63$ м, $h_H=3.75$ м, $h_B=4.81$ м, $p=45\%$ і $p=91\%$.

3.6. Бідон, який показано на рис. 15.3, має діаметр і висоту нижньої циліндричної частини d_1, h_1 відповідно; діаметр і висоту циліндричної горловини d_2, h_3; висоту конічної частини h_2. У бідон до висоти H_1 налито рідину при температурі T_1, яка має коефіцієнт об'ємного розширення α. Зміна об'єму рідини за рахунок зміни температури з T_1 до T_2 обчислюється за формулою $V_2=\alpha V_1\dfrac{T_2}{T_1}$, у якій використовуються значення температури за шкалою Кельвіна. ($T(^0K)=273+T(^0C)$). Скласти програму обчислення висоти рівня рідини H_2 при температурі $T_2 > T_1$, передбачивши в ній ознаку переливання рідини через верхній край горловини, якщо $H_2 > h_1+h_2+h_3$.

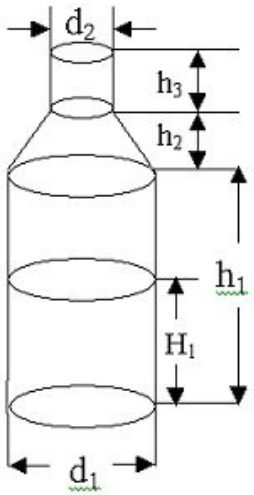


Рис. 13.4.

Контрольний варіант вхідних даних:

d_1=0.5 м, d_2=0.15 м, h_1=0.5 м, h_2=0.1 м, h_3=0.15 м, H_1=0.65 м, $T_1=10^0$C, $T_2=25^0$C, $\alpha = 0.65$.

3.7. У корівнику довжиною ***l*** і висотою ***h*** на осьовій лінії стелі встановлено три однакових світильники, один з них розташовано у центральній частині, а два інших - по обох боках від нього на відстані ***a***. Освітленість корівника на рівні підлоги на відстані ***x*** від бічної стіни уздовж його осьової лінії з урахуванням кутів падіння променів виражається формулою

$$E(x) = Ih\left\{\left[h^2+\left(\frac{l}{2}-a-x\right)^2\right]^{-3/2}+\left[h^2+\left(\frac{l}{2}-x\right)^2\right]^{-3/2}+\left[h^2+\left(\frac{l}{2}+a-x\right)^2\right]^{-3/2}\right\},$$

де ***I*** – сила світла одного світильника. Скласти програму обчислення відстані між світильниками ***a***, при якому відношення

$$K = E_{\min}/E_{\max} = \min\left[E(0), E\left(\frac{l-a}{2}\right)\right] / \max\left[E\left(\frac{l-2a}{2}\right), E\left(\frac{l}{2}\right)\right]$$

має найбільше значення. Програма повинна забезпечувати виведення значень $a, E_{\min}, E_{\max}, K$.

Контрольний варіант вхідних даних: $l = 100м, h = 4м, I = 300свічок$.

3.8. Скласти програму побудови графіка аналітично заданої функції.

3.9. Від найближчої АТС міжрайонної телефонної мережі зв'язку до правління сільгосппідприємства проведено зовнішню двухпровідну лінію зв'язку. У будинку правління встановлено місцеву АТС, яка виключає можливість одночасного підключення до зовнішньої лінії більш одного абонента. До власної АТС підключено ***N*** телефонних апаратів внутрішніх абонентів, які мають можливість виходити на зовнішню лінію зв'язку. Скласти програму обчислення припустимого середнього часу ***M*** **[хв]** однієї зовнішньої телефонної розмови, якщо середній час чекання абонентами звільнення зовнішньої лінії зв'язку не повинен перевищувати ***S*** **[хв]**. Потреба у телефонному зв'язку з абонентами за межами підприємства протягом години виникає в ***p%*** абонентів підприємства, при цьому $S = \dfrac{p^2 N^2 M^3}{6000(6000 - pNM)}$.

Контрольний варіант вхідних даних: ***N=93, p=9%, S=10 хв.***

3.10. На ділянці саду, що має форму трикутника ***ABC*** зі сторонами ***AC=a, AB=b, BC=c*** (рис. 13.4), необхідно висадити плодові дерева на однаковій відстані ***d*** одне від другого у кожному подовжньому і поперечному ряді. Перший ряд дерев висаджують уздовж сторони ***AC***, причому перше дерево саджають у вершині ***A.*** Скласти програму обчислення кількості рядів дерев ***n***, рівнобіжних стороні ***AC***, кількості дерев у кожнім ряді $m_1, m_2, ..., m_n$ і сумарного числа дерев ***k*** на ділянці. На границі саду дерева висаджувати можна. На форму саду ніякі додаткові обмеження не накладаються.

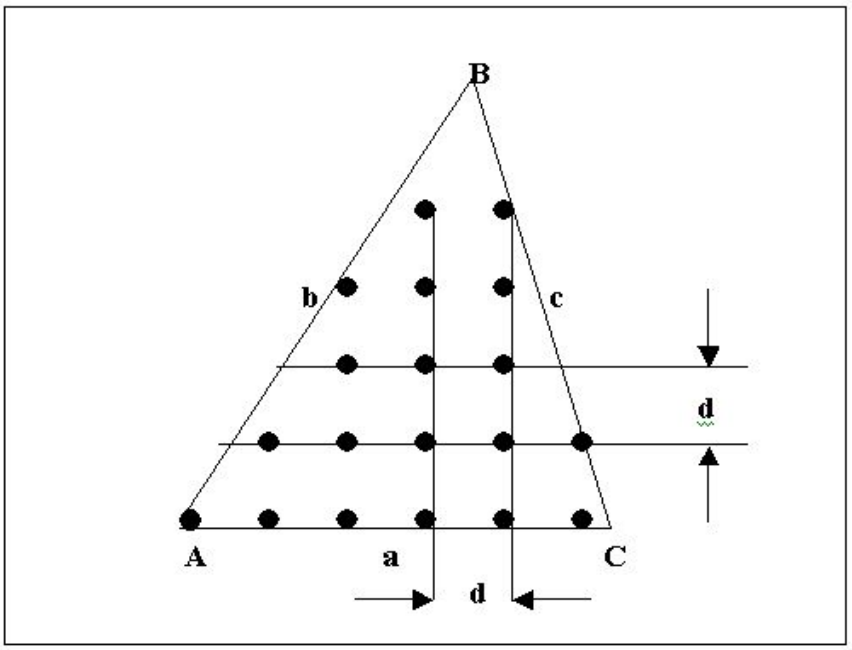


Рис. 13.4

Контрольні варіанти вхідних даних:

1) a = 150 м, b=200 м, c =170 м, d = 2.7 м;

2) a = 450 м, b=200 м, c =600 м, d = 3.8 м;

3) a = 550 м, b=700 м, c =270 м, d = 4.1 м.

3.11. Для 2-х трикутників, заданих на площині координатами своїх вершин:

а) визначити, чи мають вони загальну частину (позитивна відповідь при наявності хоча б однієї загальної точки);

б) якщо перетинанням є фігура, обчислити її площу, якщо перетинанням є відрізок, обчислити його довжину, якщо перетинанням є точка, знайти її координати;

в) зобразити трикутники на екрані (у цьому випадку передбачається, що трикутники цілком помістяться на екран) і при наявності загальної частини зафарбувати її в червоний колір.

3.12. Задане чотиризначне число і три цифри (можливе повторення цифр). Написати програму, яка знаходить різні представлення цього числа у вигляді суми тризначних чисел, складених тільки з заданих трьох цифр (складові не обов'язково повинні розрізнятися). Представлення, одержані перестановкою складових вже приведеного представлення, не наводити.

Наприклад, для числа***2003*** і цифр ***2,3,3*** існує ***10*** представлень:

2003=222+222+222+222+223+223+223+223+223

2003=222+223+223+223+223+223+333+333

2003=222+223+223+223+223+233+323+333
2003=222+223+223+223+233+233+323+323
2003=223+223+223+223+223+223+332+333
2003=223+223+223+223+223+232+323+333
2003=223+223+223+223+223+233+322+333
2003=223+223+223+223+223+233+323+332
2003=223+223+223+223+232+233+323+323
2003=223+223+223+223+233+233+322+323

Для того ж числа і цифр ***7,8,9*** шукане представлення відсутнє.

Контрольний варіант вхідних даних: ***число 1332 і цифри 1,2,3.***

3.13. Будинки у селі розташовані впродовж вулиці, по якій ходить автобус, що зв'язує село з іншими населеними пунктами (рис.13.5). Кількість будинків у селі ***m***. Відстань від воріт різних осель до краю вулиці S_1, S_2, ...,S_m,а середня (за місяць) кількість мешканців, які йдуть від цих будинків до автобусної зупинки, відповідно N_1, N_2, ..., N_m. Скласти програму обчислення оптимальної відстані ***x*** з точністю Δx від краю вулиці до автобусної зупинки, при якій загальний шлях ***L,*** який проходять мешканці села протягом місяця між своїми будинками та зупинкою автобуса, є мінімальним.

Рис.1

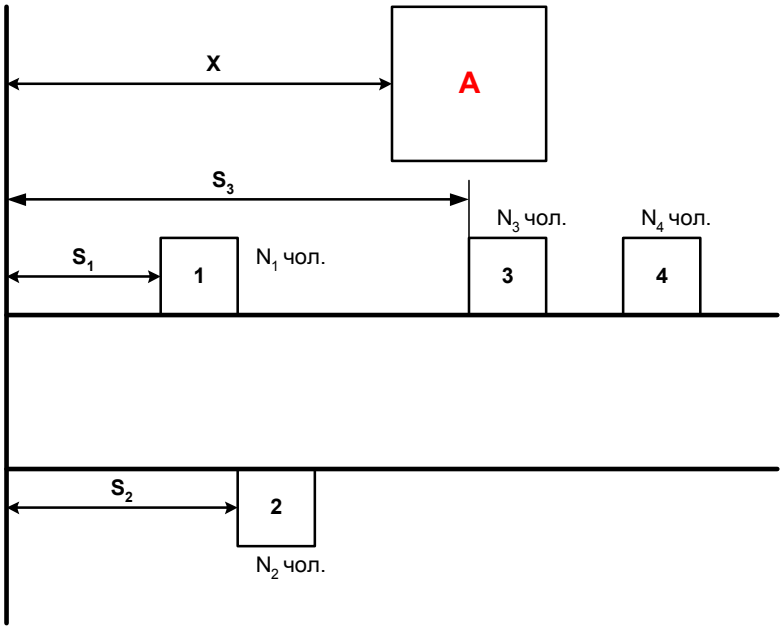


Рис.13.5

Виконати розрахунок для $\Delta x = 1$**см** за такими даними:

Номери будинків	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**
Кількість виходів мешканців до зупинки за місяць	14	17	22	16	11	19	15	8	12	19	21	13	29
Відстань від воріт осель до краю вулиці, м	23	21	45	44	92	92	133	135	176	175	232	231	285

Номери будинків	**14**	**15**	**16**	**17**	**18**	**19**	**20**	**21**	**22**	**23**	**24**	**25**	**26**
Кількість виходів мешканців до зупинки за місяць	21	9	18	15	23	16	13	12	28	19	7	16	12
Відстань від воріт осель до краю вулиці, м	286	334	335	376	380	420	423	477	479	529	527	577	579

3.14. Визначити кількість п’ятниць, які приходилися на 13-ті числа у XX столітті. Вивести місяць та рік для кожної такої п’ятниці.

3.15. Дано ціле невід’ємне число **N (N <= 4294967295)**. При записі **N** у двійковій системі числення виходить послідовність з **0** і **1**. Наприклад, при **N=19** одержуємо $\mathbf{19_{10}=1*2^4+0*2^3+0*2^2+1*2^1+1*2^0}$, тобто в двійковій системі число запишеться як $\mathbf{10011_2}$. При циклічному зрушенні **вправо** виходять інші числа. Знайти максимальне серед цих чисел. Так, для числа **19** результат циклічного зрушення наступний:

$\mathbf{10011_2=19_{10}}$

$\mathbf{11001_2=25_{10}}$

$\mathbf{11100_2=28_{10}}$

$\mathbf{01110_2=14_{10}}$

$\mathbf{00111_2=7_{10}}$

$\mathbf{10011_2=19_{10}}$.

Відповідно, максимальним є число **28**.

Привести результати роботи програми для чисел **181, 234759, 4078945139.**

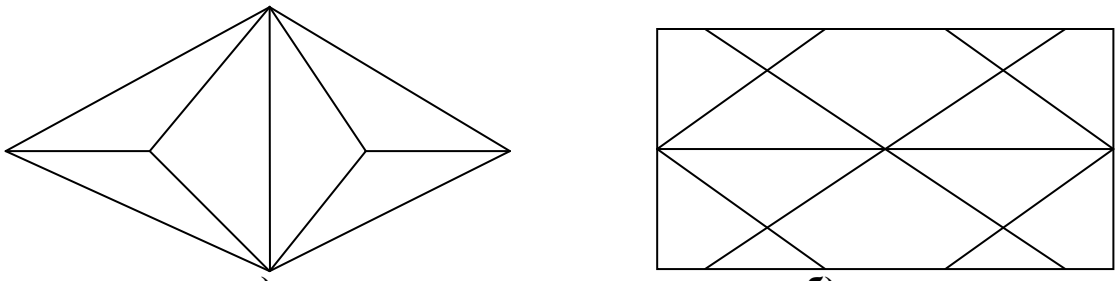
3.16. Написати програму, що обчислює частку і залишок від ділення двох цілих чисел, що не перевищують 10^{50}.

Привести результати роботи програми для

123456789098765432112345678 9:13579086421357908

3.17. На площині дані N точок, деякі з яких сполучені відрізками прямих. Написати програму, що обчислює кількість чотирикутників, що утворилися при цьому.

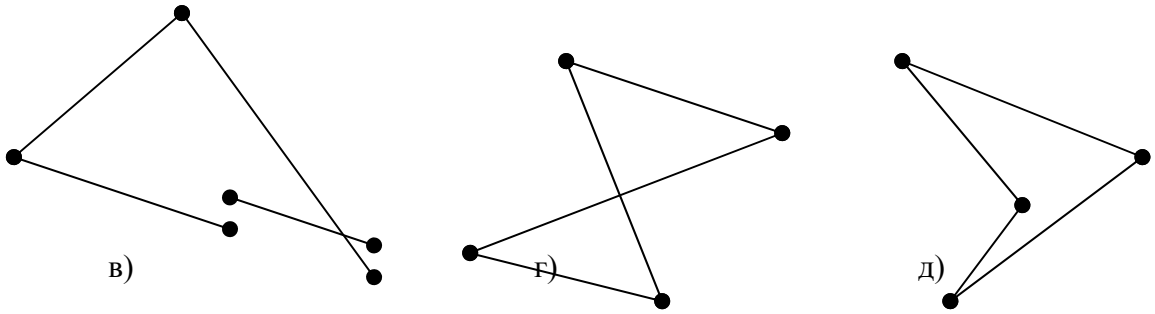
Привести результати роботи програми для приведених на рисунку двох випадків (усі точки перетину входять в початкову множину точок):



а) **б)**

Примітка:

Чотирикутником називається фігура, отримана в результаті з'єднання відрізками прямих чотирьох точок на площині, ніякі три з яких не лежать на одній прямій, і розбиваюча площину на **дві частини** – внутрішню і зовнішню. Відповідно до цього визначення фігури в) і г) не є чотирикутниками, а фігура д) – чотирикутник (не опуклий).



в) г) д)

3.18. Таблиця футбольного чемпіонату задається квадратною матрицею ***A*** розмірності ***n***, у якій всі елементи, які належать головній діагоналі, дорівнюють нулю, а кожен елемент, який не знаходиться на головній діагоналі, може дорівнювати або ***2***, або ***1***, або ***0*** (в залежності від кількості виграних балів: 2 – виграш, 1 – нічия, 0 – програш). Наприклад, елемент

матриці, який знаходиться на ***5*** рядку у ***1***-му стовбці $a_{51}=0$, це означає, що команда ***№ 5*** програла команді ***№ 1***, значить, в свою чергу команда ***№ 1*** виграла у команди ***№ 5,*** тобто $a_{15}=2$.

Необхідно знайти:

Кількість команд, які мають більше перемог, ніж поразок;

Визначити номери команд, які пройшли чемпіонат без поразок;

З'ясувати чи є принаймні одна команда, яка виграла більше половини ігор.

		Номери команд						
		1	**2**	**3**	**4**	**5**	**6**	**7**
Номери команд	**1**	0	2	1	0	2	0	0
	2	0	0	2	1	2	1	1
	3	1	0	0	0	1	0	2
	4	2	1	2	0	2	0	1
	5	0	0	1	0	0	1	2
	6	2	1	2	2	1	0	1
	7	2	1	0	1	0	1	0

3.19. Знайти в **n**-вимірному просторі мінімальну відстань від початку координат до відрізку АВ, заданого координатами своїх кінців.

Привести результати роботи програми для:

n=4;

а) A(2,15,4,-7), B(8,3,5,-3);

б) A(2,15,-2,6), B(4,10,-5,7).

(для n=2 та n=3 , взяти перші координати точок).

3.20. Необхідно реалізувати таку (дуже спрощену) демографічну модель. Початкове населення держави становить **N** людей однакового віку. Тривалість життя у всіх теж однакова і становить **L** років. У віці **Z** кожне покоління народжує дітей з коефіцієнтом відтворення **k** на кожну сімейну пару, тобто населення в цей період відразу ж збільшиться на **k*N/2** (вважаємо, що в державі завжди підтримується чисельна рівність статей і всі створюють сім'ї ☺). Розрахувати по роках за вказаний період кількість населення.

Всі розрахунки можна вести с плаваючою комою без округлення, на екран виводити цілі числа.

Привести результати зміни чисельності населення за **500 років** для двох держав. У першій державі за рахунок високого рівня розвитку економіки, культури і медицини тривалість життя всіх громадян становить **L=200 років**, однак коефіцієнт відтворення лише **1,5**. У другій державі, у зв'язку з низьким рівнем життя, мешканці живуть всього **45 років**, проте коефіцієнт відтворення є відносно високим – **3,0**. У перший рік кожна держава має **1 млн.** громадян у віці **20 років**, ***Z*=25 років**.

3.21. Розглянемо такий алгоритм генерування послідовності чисел. Почнемо з цілого числа ***n.*** Якщо число **парне (“четное”)**, то розділимо його на **2**, якщо ні, то помножимо на **3** та додамо **1**. Будемо повторювати цей процес із новим отриманим ***n***, до тих пір, поки ***n*** не стане дорівнювати 1. Наприклад, для ***n=22*** буде згенеровано таку послідовність чисел:

22 11 34 17 52 26 13 40 20 10 5 16 8 4 2 1

Вважають (але це ще не доведено), що алгоритм буде зведено до ***n=1*** для будь-якого цілого ***n***. Принаймні, ця пропозиція вірна для усіх цілих чисел до 1000000.

Для даного ***n*** довжиною циклу називатимемо число згенерованих чисел, включаючи саме число та 1. У наведеному прикладі довжина циклу дорівнює 16.

Програма для двох заданих позитивних чисел, що не перевищують $\mathbf{10^9}$, повинна обчислювати максимальну довжину циклу серед усіх чисел між ними, включаючи самі ці числа.

Навести результати роботи програми для інтервалу **[9900000; 10000000]**.

14. РОЗПОДІЛ МАТЕРІАЛУ НА МОДУЛІ

Матеріал методичних вказівок вивчають на першому курсі у двох модулях.

МОДУЛЬ V. Тема «Мова програмування Сі (частина І)»

Лекції – 3 год., лабораторних занять – 8 год., самостійна робота – 6 год.

Розділи 1-9, 12, 13.

Розподіл балів по пунктах контролю:

Лабораторна робота № 5. «Розрахунок контрольних варіантів для індивідуальної задачі з програмування» – **10 балів.**

Лабораторна робота № 6. «Написання тексту програми». Розв'язання індивідуальної задачі за сільськогосподарською тематикою. Складання тексту програми – **10 балів.**

Контрольна робота по основним операторам – **5 балів.**

Лабораторна робота № 7. «Введення програми на ПК і налагодження за контрольними варіантами». Набір тексту програми на ПК та її налагодження за допомогою середовища програмування С++ Builder6 – **10 балів.**

Всього за модуль – **35 балів.**

МОДУЛЬ VI. Тема «Мова програмування Сі (частина ІІ)»

Лекції – 7 год., лабораторних занять – 8 год., самостійна робота – 6 год.

Розділи 10-12, 12, 13.

Розподіл балів по пунктах контролю:

Лабораторна робота № 1. «Доопрацювання тексту індивідуальної програми для введення даних з клавіатури». Відлагодження програми за контрольним прикладом. Доопрацювання текстів програм для реалізації введення даних з клавіатури. – **15 балів.**

Лабораторна робота №2. «Введення програми на ПК і налагодження за контрольними варіантами». Налагодження програм на ПК та перевірка працездатності за різними варіантами вхідних даних. – **15 балів.**

Разом за модуль – **30 балів.**

СПИСОК ЛІТЕРАТУРИ

1. Уэйт М. Язык Си. Руководство для начинающих / М. Уэйт, С. Прата, Д. Мартин. – М. : Мир, 1988. – 237 с.
2. Трой Д. Программирование на языке Си для персональных компьютеров IBM PC / Д. Трой. – М. : Радио и связь, 1991. – 386 с.
3. Белецкий Я. Энциклопедия языка Си / Я. Белецкий. – М. : Мир, 1992. – 496 с.
4. Керниган Б. Язык программирования Си / Б. Керниган, Д. Ритчи. – М. : Финансы и статистика, 1992. – 396 с.
5. Дьюхарст К. Программирование на Си++ / К. Дьюхарст, К. Старк. – К. : НИПФ "ДиаСофт", 1993. – 87 с.
6. Лукас П. Си++ под рукой / П. Лукас. – К. : НИПФ "ДиаСофт", 1993. – 143 с.
7. Савельев А. Я. Задачи и упражнения по программированию / А. Я. Савельев. – М., 1989. – 54 с.
8. C++ Builder 5. Руководство разработчика. – В 2-х ч. – Ч. 1. / [Джарод Холингвэрт, Ден Баттерфилд и др.]. – М. ; СПб ; К., 2001. – 1638 с.
9. C++ Builder 5. Руководство разработчика. – В 2-х ч. – Ч. 2. / [Джарод Холингвэрт, Ден Баттерфилд и др.]. – М. ; СПб ; К., 2001. – 1638 с.

Навчальне видання

ІНФОРМАЦІЙНІ СИСТЕМИ ТА ТЕХНОЛОГІЇ

Методичні рекомендації

Укладачі : **Іхсанов** Шаміль Мухаметович,
Лопушанська Валентина Володимирівна

Формат 60х84/16. Ум. друк. арк. 5,4.

Тираж 50 прим. Зам. №___

Надруковано у видавничому відділі

Миколаївського національного аграрного університету

54020 м. Миколаїв, вул. Паризької комуни, 9

Свідоцтво суб'єкта видавничої справи ДК №4490 від 20.02.2014 р.