Центральноукраїнський національний технічний унiверситет

Кафедра програмування комп’ютерних систем і мереж

Дисципліна: Базові методології та технології програмування

Звiт

з лабораторної роботи №5

Тема: Технологія програмування розгалужених алгоритмів

Мета роботи: Навчитися створювати алгоритми та структурну програму розрахунку функції за певної умови, використовуючи для цього умовний оператор if та оператор вибору мовами програмування Pascal та C.

Виконав: ст. гр. КМ-20

Калиновський В.М.

Перевірила: асистент кафедри ПКСМ

Тесленко О.Є.

Варіант 3

Дата виконання: 26.10.2020

Дата оформлення: 27.10.2020

**Кропивницький 2020 р.**

**Завдання:** Перекласти програму, яка дає змогу користувачу зробити розрахунки різних видів комунальних платежів, на мову С; Написати програму на двох мовах, що буде розраховувати V1 або V2 залежно від X, Y та Z, які у свою чергу залежать від C та D; Написати програму на двох мовах з використанням Switch та Case відповідно свого варіанту.

**Робота.** Для виконання першого завдання нам дали код програми розрахунку комунальних платежів на мові Паскаль. Завданням було перекласти цю програму на мову С. Для його виконання треба було спочатку розібрати алгоритм. До нього входило:

1. Виведення списку всіх можливих розрахунків
2. Введення номера виду розрахунку
3. Виклик функції розрахунку комунального платежа
   1. Передача у функцію вибраного виду
   2. Порівняння з кейсами (у випадку співпадіння починається діалог з користувачем)
      1. Запитується змінна, яка використовується у розрахунку (наприклад, кількість використаної холодної води)
      2. Підставляється значення у формулу й проводиться розрахунок
      3. Виводиться результат
4. Запитується у користувача, чи хоче він продовжити роботу
   1. У разі згоди повторюється цикл (повернення до першого пункту)
   2. У разі відмови програма завершується

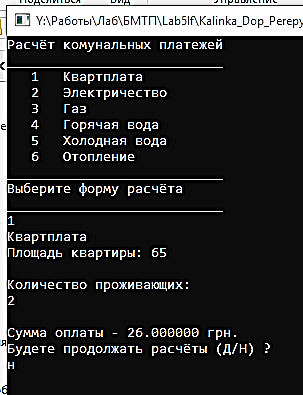
Код програми написано у Таблиці 1, а результати виконання цієї програми відображено на рисунку 1.

Таблиця 1. Код програми розрахунку комунальних платежів

|  |
| --- |
| #include <stdio.h>  #include <Windows.h>  #include <stdlib.h>  void categer (c); //Объявляю функцию  void main () {  SetConsoleCP(1251);  SetConsoleOutputCP(1251);    char c, v;    for (;;) {  printf("Расчёт комунальных платежей\r\n");  printf("\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\r\n");  printf(" 1 Квартплата \r\n");  printf(" 2 Электричество \r\n");  printf(" 3 Газ \r\n");  printf(" 4 Горячая вода \r\n");  printf(" 5 Холодная вода \r\n");  printf(" 6 Отопление \r\n");  printf("\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\r\n");  printf("Выберите форму расчёта \r\n");  printf("\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\r\n");    /\*Когда считывается первый символ в потоке,  остаётся символ перевода строки и этот символ  хавается следующей переменной. Функция fflush(stdin)  очищает поток от лишних символов, поэтому когда  вызывается следующая scanf() есть возможность  прочитать следующий символ.\*/  fflush(stdin);  scanf("%c", &c);    categer (c);    printf("Будете продолжать расчёты (Д/Н) ?\r\n");  fflush(stdin);  scanf("%c", &v);  switch (v) {  case 'y':  exit(0); //Объявляю выход в ОС  case 'н':  exit(0);  }  }  }  void categer (c){  int i;  float p, a, b, r, w, t;    switch (c) {  case '1':  printf("Квартплата\r\nПлощадь квартиры: ");  scanf("%f", &p);  printf("\r\nКоличество проживающих: \r\n");  scanf("%i", &i);  float ip = i \* p \* 0.2;  printf("\r\nСумма оплаты - %f грн.\n", ip);  break;  case '2':  printf("Электричество\r\nКоличество кВт/час, использованных за месяц: ");  scanf("%f", &a);  a = a \* 0.09;  printf("\r\nСумма оплаты - %f грн. \r\n", a);  break;  case '3':  printf("Газ\r\nКоличество кубометров, использованных за месяц: ");  scanf("%f", &b);  b = b \* 1.10;  printf("\r\nСумма к оплате - %f грн. \r\n", b);  break;  case '4':  printf("Горячая вода\r\nКоличество кубометров, использованных за месяц: ");  scanf("%f", &r);  r = r \* 1.79;  printf("\r\nСумма к оплате - %f грн. \r\n", r);  break;  case '5':  printf("Холодная вода\r\nКоличество кубометров, использованных за месяц: ");  scanf("%f", &w);  w = w \* 0.43;  printf("\r\nСумма к оплате - %f грн. \r\n", w);  break;  case '6':  printf("Отопление\r\nОбщая площадь квартиры: ");  scanf("%f", &t);  t = t \* 0.42;  printf("\r\nСумма к оплате - %f грн. \r\n", t);  break;  }  } |

Перевіримо результат: Виберемо розрахунок квартплати. У нас у квартирі проживає дві людини, а площа квартири 65. Підставимо значення: 2\*65\*0.2=130/5=26. Ця відповідь співпадає з відповіддю програми, отже вона працює правильно.

Рисунок 1. Результат програми розрахунку комунальних платежів

****

Наступна програма має бути написана на двох мовах і розраховувати значення V1 або V2 залежно від попередніх значень. Згідно мого варіанта програма має розрахувати V1, коли Y < Z, або V2 коли Y >= Z. Як отримати Y та Z написано у таблиці 2.

Таблиця 2. Формули розрахунку Y та Z

|  |
| --- |
| (1)  (2)  (3)  (4)  (5) |

З умови, C = 0.2, а D = 0.1. Спробуємо провести розрахунок: Підставляємо значення C та D у формулу 3 і отримаємо X = 0.1; Підставляємо значення D та X у формулу 1 і отримаємо Y = 0.0999; Підставляємо значення Y, C та X у формулу 2 і отримаємо Z = 1.064; Так як 0,1 < 1.064, то й Y < Z => тоді ми використовуємо 4 формулу й знаходимо V1 = 0.2. Так як я округлював результати, мої результати можуть бути відмінні від результатів комп’ютера на декілька десятих. Якщо різниця більше, то хтось неправильно рахує – чи я, чи програма (але скоріше програма). Для перевірки результату я пишу програму (Табл3-4) й отримую результат (Рис2), який більше пари сотих чи однієї десятої від моїх результатів не відрізняється. Отже програми написані правильно.

Таблиця 3. Код програми розрахунку V1 або V2, залежно від результатів на мові С

|  |
| --- |
| #include <stdio.h>  #include <Windows.h>  #include <Math.h>  int main() {  SetConsoleCP(1251);  SetConsoleOutputCP(1251);    double c = 0.2;  double d = 0.1;  double x, y, z, v;  x = c - d;  y = d \* (cos(x) \* cos(x) \* cos(x));  double yinc = pow(y, c + 1);  z = yinc + (x / y);  if (y < z) {  v = x + y;  printf("Входные данные: C = %lf; D = %lf. Выходные данные: X = %lf; Y = %lf; Z = %lf; так как Y < Z, то V1 = %lf.\n", c, d, x, y, z, v);  }  else {  v = pow(y, 2 \* z);  printf("Входные данные: C = %lf; D = %lf. Выходные данные: X = %lf; Y = %lf; Z = %lf; так как Y >= Z, то V2 = %lf.\n", c, d, x, y, z, v);  }  system("pause");  return 0;  } |

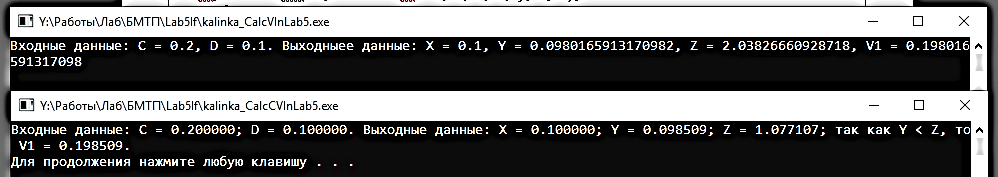
Таблиця 4. Код програми розрахунку V1 або V2, залежно від результатів на мові Паскаль

|  |
| --- |
| **program** kalinka\_CalcVInLab5;  **uses** crt;  **var** c,d,x,y,z,v:real;  **begin**  c:=0.2; d:=0.1;  x:=c-d;  y:=d\*(sqr(cos(x)\*cos(x)));  z:=(exp(y\*ln(c+1)))+(x/y);  **if** (y<z) **then** v:=x+y **else** v:=exp(y\*ln(2\*z));  **if** (y<z) **then** writeln('Входные данные: C = ', c, ', D = ', d, '. Выходныее данные: X = ', x, ', Y = ', y, ', Z = ', z, ', V1 = ', v) **else** writeln('Входные данные: C = ', c, ', D = ', d, '. Выходныее данные: X = ', x, ', Y = ', y, ', Z = ', z, ', V2 = ', v);  readkey;  **end**. |

Алгоритм цієї програми дуже простий:

1. C = 0.2; D = 0.1
2. Розрахунок X, Y та Z в заданій послідовності
3. Розрахунок V1 або V2 залежно від значень Y та Z
4. Виведення значень C, D, X, Y,Z та V1/V2 залежно від того, хто був порахований

Рисунок 2. Результати виконання даних програм

****

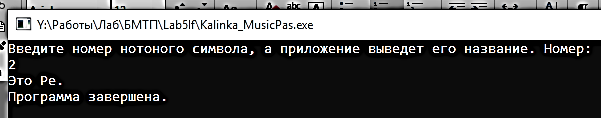
У наступному завданні я маю написати програму, яка буде питати у користувача номера ноти, а програма буде виводити її назву. На Паскалі я не став заморочуватись з перевіркою натиснутих клавіш та з виходом із програми. Отримав наступну програму (код знаходиться у таблиці 5 а результат на рисунку 3):

1. Програма питає номер ноти
2. Коли номер задано:
   1. Неправильний – йде завершення програми
   2. Правильний:
      1. Виводиться назва ноти
      2. Йде завершення програми

Таблиця 5. Програма, що виводить назву ноти на Паскалі

|  |
| --- |
| **Program** Kalinka\_Music;  **uses** crt;  **var** NomMuzZnak:integer;  **begin**  writeln('Введите номер нотоного символа, а приложение выведет его название. Номер: ');  readln(NomMuzZnak);  **case** NomMuzZnak **of**  1:writeln('Это До.');  2:writeln('Это Ре.');  3:writeln('Это Ми.');  4:writeln('Это Фа.');  5:writeln('Это Соль.');  6:writeln('Это Ля.');  7:writeln('Это Си.');  **end**;  writeln('Программа завершена.');  readkey;  **end**. |

Рисунок 3. Результат виконання музичної програми на Паскалі

****

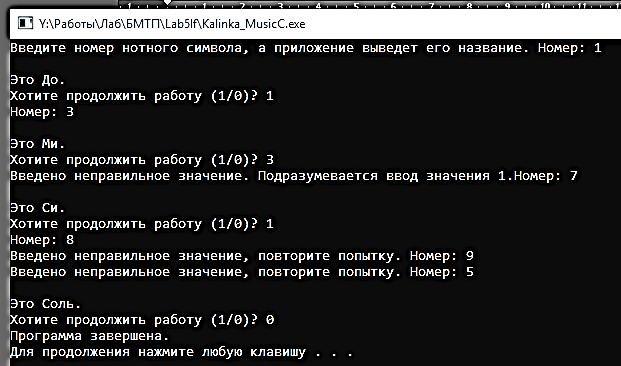
Для такої ж програми на мові С я вирішив заморочитись й добавити перевірку введених символів та вихід з програми (код написано у таблиці 6, а результат на рисунку 4). Працює вона наступним методом:

1. Програма просить ввести номер ноти
2. Коли номер задано неправильний, програма починається с початку
3. Коли номер задано правильний, програма виводить результат и викликає функцію виходу:
   1. Функція питає, чи хоче користувач продовжити роботу
   2. Якщо відповідь негативна – програма завершається
   3. Якщо неправильна – приймається як за позитивну
   4. Якщо позитивна – починається спочатку

Таблиця 6. Код музичної програми на мові С

|  |
| --- |
| #include <stdio.h>  #include <Windows.h> //Подключаю библиотеку с функциями настройки ввода-вывода  boolean ProverNomMuzZnak = 0; //Переменная, отвечающая за выход из цикла общения с пользователем; объявляю её глобальной  void FProverVyhod();  void main() {  SetConsoleCP(1251); //Настраиваю ввод  SetConsoleOutputCP(1251); //Настраиваю вывод  int NomMuzZnak; //Переменная номера музыкального знака  boolean Text = 0; //Переменная контролирующая цикл 16-18 строк  printf("Введите номер нотного символа, а приложение выведет его название. Номер: ");  do {  /\*Цикл if (Text == 1) {} не работает когда программа только запустилась, однако  при дальнейшем использовании программы этот цикл будет всегда работать\*/  if (Text == 1) {  printf("Номер: ");  }  Text = 1; //Включает цикл на 16-18 строчках  fflush(stdin); //Обнуляем входной поток  scanf("%i", &NomMuzZnak);  /\*Следующий Switch отвечает на ответ от пользователя. Его цель: вывести ответ и спросить, не хочет ли  пользователь продолжить работу. В случае: неправильного ввода в начале, цикл начинается заново; если  неправильно введено значение при ответе на продолжение работы, выбирается положительный ответ и цикл  повторяется; при положительном ответе на продолжение работы цикл повторяется; и только при негативном  ответе на продолжение работы цикл завершается. Этот процес регулируется булевыми переменными\*/  switch (NomMuzZnak) {  case 1:  printf("\nЭто До.\nХотите продолжить работу (1/0)? "); //Выводим результат и спрашиваем продолжение работы  //Дальше следует ввод ответа и проверка введёного ответа - для этого вызывается функция FProverVyhod  FProverVyhod();  break;  /\*Следующие кейсы от 2 до 7 аналогичны 1 за тем лишь отличием, что в строчке вывода будет  отличаться выходной результат (не До, а, например, Ре или Ля)\*/  case 2:  printf("\nЭто Ре.\nХотите продолжить работу (1/0)? ");  FProverVyhod();  break;  case 3:  printf("\nЭто Ми.\nХотите продолжить работу (1/0)? ");  FProverVyhod();  break;  case 4:  printf("\nЭто Фа.\nХотите продолжить работу (1/0)? ");  FProverVyhod();  break;  case 5:  printf("\nЭто Соль.\nХотите продолжить работу (1/0)? ");  FProverVyhod();  break;  case 6:  printf("\nЭто Ля.\nХотите продолжить работу (1/0)? ");  FProverVyhod();  break;  case 7:  printf("\nЭто Си.\nХотите продолжить работу (1/0)? ");  FProverVyhod();  break;  default:  /\*Дефолт служит на случай неправильного ввода ответа на выбор ноты. Он сразу готовит цикл на  повтор, чтобы пользователь мог повторно ввести данные\*/  ProverNomMuzZnak = 1; //установка повторения цикла  printf("Введено неправильное значение, повторите попытку. ");  break;  }  } while (ProverNomMuzZnak == 1); //условие повторения цикла (не выполняется только в случае отказа на продолжение работы)  //В случае отказа от продолжения работы цикл завершается и следует завершение программы  printf("Программа завершена.\n");  system("pause");  }  void FProverVyhod() {  int ProverVyhod; //Переменная проверки пользовательской инфо на ответ про выход  fflush(stdin); // обнуляем входной поток  scanf("%i", &ProverVyhod); //Пользователь вводит ответ на вопрос про выход  //Дальше следует проверка введёного ответа  switch (ProverVyhod) {  case 0:  ProverNomMuzZnak = 0; //Завершаем цикл  break;  case 1:  ProverNomMuzZnak = 1; //Повторяем цикл  break;  default:  printf("Введено неправильное значение. Подразумевается ввод значения 1.");  ProverNomMuzZnak = 1; //Повторяем цикл  break;  }  } |

Рисунок 4. Результат виконання музичної програми на мові С

****

Всі нюанси й подробиці розписано у коментарях у програмі. Блок-схема цієї програми буде в кінці звіту після контрольних запитань.

Відповіді на контрольні питання:

1. Які оператори мови програмування називаються структурованими?

if, if … else, switch … case (можна звести до if);

while, do … while, for (можна звести до while)

1. Що таке вкладений оператор умови та які функції він виконує? Навести приклад.

Вкладений оператор умови – оператор, що контролює виконання циклу. Виконує функцію перевірки значень - у разі відповіді true оператор повторює цикл, а у разі відповіді false завершає цикл, наприклад, цей цикл виведе фразу ««Hello world»» п’ять разів:

x=1

do {

printf(«Hello world!»);

x++;

} while (x<5);

1. Як працює умовний оператор IF (обидві мови)?

IF схожий на вкладений оператор умови. Він теж виконує функцію перевірки значень але у разі відповіді true оператор виконує вкладені у нього функції, а у разі відповіді false виконує наступні «else if» (якщо їх значення співпадають) та «else» якщо жоден з if не виконується; у разі відсутності else та невиконанні жодного з if – цей оператор взагалі не виконується та пропускається

1. Чи може символ ; зустрічатись в структурованому операторі й чому?

Він зустрічається у циклі for для відділення лічильника, умови та функції друг від друга, наприклад: «*for (i=1; i<10; i++);*» або просто «*for (;;);*»

1. Які особливості при написанні вкладених операторів IF?

Кожен оператор починається з нової строки й має закінчуватись «;»

1. Написати код (завдання написано у самій лабораторній роботі).

Код розв’язку завдання A написано у таблиці 7, B у 8 та C у 9, а результати на рисунку 6 .

Таблиця 7. Рішення завдання A

|  |
| --- |
| #include <stdio.h>  #include <math.h>  #include <Windows.h>  int x, y;  void f(int x) {  if (x < -5) {  y = x - 1;  printf("\nПри X = %i: Y = %i\n", x, y);  }  else if (x > 5) {  y = x \* x + 2;  printf("\nПри X = %i: Y = %i\n", x, y);  }  else {  printf("\nРешение отстутствует.\n");  }  }  void main() {  SetConsoleCP(1251);  SetConsoleOutputCP(1251);  printf("Введите значение X: ");  scanf("%i", &x);  f(x);  system("pause");  } |

Таблиця 8. Рішення завдання B

|  |
| --- |
| #include <stdio.h>  #include <math.h>  #include <Windows.h>  int x, y;  void f(int x) {  if (x > -20 && x < 0) {  y = x \* x \* x - 2 \* x;  printf("\nПри X = %i: Y = %i\n", x, y);  }  else if (x > 5) {  y = x - x \* x;  printf("\nПри X = %i: Y = %i\n", x, y);  }  else {  printf("\nРешение отстутствует.\n");  }  }  void main() {  SetConsoleCP(1251);  SetConsoleOutputCP(1251);  printf("Введите значение X: ");  scanf("%i", &x);  f(x);  system("pause");  } |

Таблиця 9. Рішення завдання C

|  |
| --- |
| #include <stdio.h>  #include <math.h>  #include <Windows.h>  float x, y;  void f(int x) {  if ((x > -15 && x < 1) || (x > 15 && x < 30)) {  y = x / 2. + 5;  printf("\nПри X = %i: Y = %f\n", x, y);  }  else if ((x > 5 && x < 10) || (x > 40)) {  y = (x \* x) + (1. / x);  printf("\nПри X = %i: Y = %f\n", x, y);  }  else {  printf("\nРешение отстутствует.\n");  }  }  void main() {  SetConsoleCP(1251);  SetConsoleOutputCP(1251);  printf("Введите значение X: ");  scanf("%f", &x);  f(x);  system("pause");  } |

1. Написати код (завдання написано у самій лабораторній роботі).

Код розв’язку завдання 7 написано у таблиці 10, результат на рисунку 6 .

Таблиця 10. Розв’язок сьомого завдання

|  |
| --- |
| #include <stdio.h>  #include <math.h>  #include <Windows.h>  float x, y;  void f(float x) {  if (x < 0) {  y = x \* x + 1;  printf("\nПри X = %i: Y = %.3f\n", x, y);  }  else {  y = 1.7 \* x \* x \* x + fabs(cos(x) - 2.1);  printf("\nПри X = %i: Y = %.3f\n", x, y);  }  }  void main() {  SetConsoleCP(1251);  SetConsoleOutputCP(1251);  printf("Введите значение X: ");  scanf("%f", &x);  f(x);  system("pause");  } |

1. Написати код (завдання написано у самій лабораторній роботі).

Код розв’язку завдання 7 написано у таблиці 11, результат на рисунку 6 .

Таблиця 11. Розв’язок восьмого завдання

|  |
| --- |
| #include <stdio.h>  #include <math.h>  #include <Windows.h>  int a, b, c, d;  float x1, x2;  void f(int a, int b, int c) {  if (a < 0) {  printf("Уравнение вырожденное.\n");  }  else {  d = (b \* b) - (4 \* a \* c);  if (d < 0) {  printf("Действительных корней не существует.\n");  }  else if (d == 0) {  x1 = -b / (2 \* a);  printf("Так как D = 0, => X1 = X2 = %.3f\n", x1);  }  else if (d > 0) {  x1 = (-b + sqrt(d)) / (2 \* a);  x2 = (-b - sqrt(d)) / (2 \* a);  printf("X1 = %.3f; X2 = %.3f\n", x1, x2);  }  }  }  void main() {  SetConsoleCP(1251);  SetConsoleOutputCP(1251);  printf("Решение квадратных корней.\n");  printf("Введите значение A: ");  scanf("%i", &a);  printf("Введите значение B: ");  scanf("%i", &b);  printf("Введите значение C: ");  scanf("%i", &c);  f(a, b, c);  system("pause");  } |

1. Який оператор дозволяє виконати одну із декількох дій в залежності від результату обчислення виразу?

IF

1. Як працює оператор CASE та SWITCH?

CASE зберігає значення змінної, яке необхідне для виконання певних функцій, а SWITCH приймає надане значення значення й перевіряє його зі встановленими збереженими значеннями. Це використовується для перевірки нажатих клавіш й замінює оператор IF, так як вважається більш зручним для цієї мети. Наприклад: користувач натискає клавішу Х, це значення надається SWITCH, який перевіряє це значення на співпадіння зі всіма кейсами – якщо співпаде з кимось – виконується функція, вкладена у ций CASE, якщо ні з ким не співпала, виконується DEFAULT, якщо DEFAULT не прописано, цей оператор повністю пропускається й не виконується

1. Чим відрізняється оператор CASE від інших умовних операторів?

Оператор CASE більш схожий на звичайну змінну, яка зберігає певне значення

1. Що називається селектором?

Вся структура коду називається «набором правил». Цей набір поділяється на селектори, об’яви, властивості та значення властивостей. Селектори – імена елементів на початку набора правил. Наприклад, селектором буде – main, об’явою буде – include, властивістю буде – printf та значенням властивості буде – «Hello world!» у коді:

#include <stdio.h>

void main() {

printf(«Hello world!»);

}

1. Що називається констатною вибору? Чи може водночас значення селектора дорівнювати (‘2’), а значення константного вибору (2)? Пояснити.

Константна вибору – одне із можливих значень CASE. Одночасно значення селектора дорівнювати (‘2’), а значення константного вибору (2) може, але перше значення є char, а друге є int, тому при переведенні цих значень у машинний код вони будуть закодовані по різному й значення цього виразу виконуватись не буде.

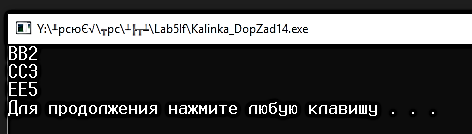
1. Які повідомлення виведуться у фрагменті програми (Табл12)? Пояснити.

Результат на рисунку 5. Такий результат отримався, оскільки A != 3, 3<6 та -4<6

Таблиця 12. Що виведеться у результаті цієї програми?

|  |
| --- |
| #include <stdio.h>  void main() {  int x = 5;  int y = 6;  int z = 10;  if (x == 3) puts("AA1");  else puts("BB2");  if (x = 3, x < y) puts("CC3");  else puts("DD4");  if (z < y, z = -4) puts("EE5");  else puts("FF6");  system("pause");  } |

Рисунок 5. Результат програми



1. Які значення може приймати вираз в операторі SWITCH(вираз)?

int та char

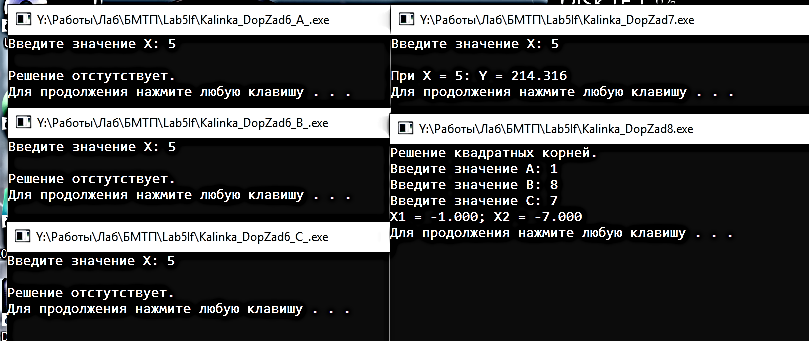
1. Яке призначення мітки default у операторі множинного вибору? Чи завжди вона потрібна? Що станеться при її вилучені?

Призначена для виконання вкладеного у нього виразу в разі жодного неспівпадіння. Потрібна не завжди. У разі її вилучення й при жодному неспівпадінні SWITCH просто пропускається й не виконується

1. Яке призначення оператора break у операторі множинного вибору? Чи завжди він потрібен? Що станеться при його вилучені?

Він призначений для преривання й пропуску SWITCH у разі співпадіння й виконання одного з кейсів. Потрібен не завжди. При його вилученні просто перестане виконуватись преривання й пропуск SWITCH й після виконання вкладених у кейс функцій перевірка продовжиться.

Рисунок 6. Результат виконання програм 6A, 6B, 6C, 7 та 8



Блок-схема до таблиці 6 знаходиться на наступній сторінці.