Лабораторно упражнение № 5

Тема: Управляващи оператори в езика C++. Реализиране на разклонени алгоритми.

I. Цел на лабораторното упражнение

Да се затвърдят знанията на студентите за организацията и синтаксиса на програми написани на програмният език С. Да се усвои работата с интегрирани среди за програмиране

II. Постановка на задачата

Много често се налага да избираме между няколко възможни варианта за решаването на дадена задача. Какво решение ще вземем обикновено зависи от някакво условие. Това условие определя как ще продължи изпълнението на програмата. За да бъде изпълнена тази операция в програмните езици са въведени група оператори които под общото наименование (оператори за условно разклонение). В тази група спадат операторите `if .. else` и `switch case .. default`. `(?:)`

III. Теоретични сведения

Оператори за избор

Операторите за избор приемат решение на базата на заключението от даден логически израз, който дава заключение истина или лъжа. Логическите изрази могат да се образуват на базата на операции, равенства и отношения

Таблица 1 – Логически изрази в C++

Операция в C++	Условие	Коментар
==	a == b	a е равно на b
!=	a != b	a не е равно на b
>	a > b	a е по-голямо от b
<	a < b	a е по-малко от b
>=	a >= b	a е по-голямо или равно на b
<=	a <= b	a е по-малко или равно на b

Много често грешките в програмите се дължат на неправилно записване на логическите отношения. Интервалите между символите в операциите `==`; `!=`; `>=`; `<=`; водят до грешки.

Условен оператор if else

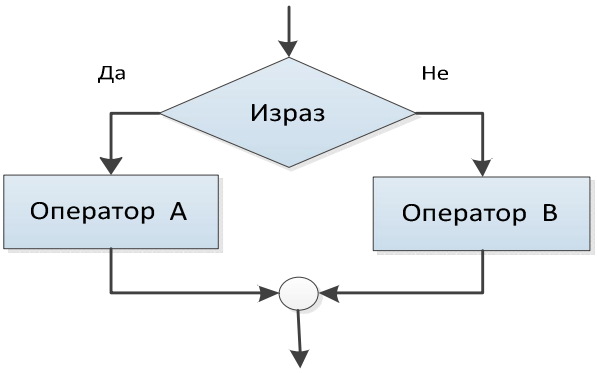
Условен оператор се използва, когато се налага да се направи избор. Изборът може да бъде между две алтернативни възможности или да се избира да се изпълни ли даден оператор или да не се изпълни.

В зависимост от вида на избора в езика C/C++ се използват два варианта на условен оператор: с пълен и непълен формат.

Условен оператор в пълен формат (if else).

Този оператор може да се представи в логическа форма посредством изречението:

Ако Израз тогава Оператор A иначе Оператор B;



Фиг.1. Условен оператор в пълен формат

Тук **Израз** е логически израз (има логическа стойност), **Оператор A** и **Оператор B** са оператори, които могат да бъдат прости или съставни (блокове). Схемата на тази конструкция е показана на фиг. 1. Записан като програмен оператор на C/C++, той има вида:

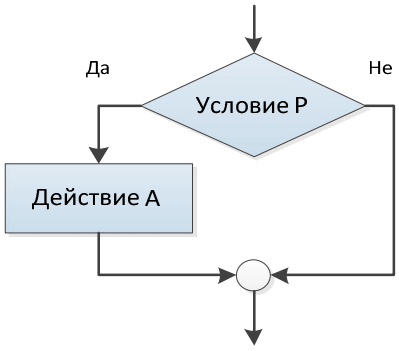
```
if( Израз ) {
  Оператор A
} else {
  Оператор B;
}
```

Действието на условния оператор в пълен формат е следното: проверява се стойността на логическия израз **Израз**; ако стойността е истина (**true**), се изпълнява **Оператор A**, а ако стойността е неистина (**false**) - се изпълнява оператор **Оператор B**.

Условен оператор в непълен формат (if).

Може да се представи в логическа форма посредством изречението:

Ако Израз тогава Оператор A;



Фиг.2. Условен оператор в кратък формат

Стойността на логическия израз **Израз** определя дали да се изпълни оператора **Оператор A** или да не се изпълни. Ако стойността на **Израз** е истина (**true**), операторът **Оператор A** ще бъде изпълнен, а ако е лъжа (**false**), няма да бъде изпълни. Схемата на тази синтактична конструкция е показана на фиг. 2. Като програмен оператор на C/C++ той има видът:

```
if( Израз ) {
   Оператор A
}
```

Вложение на условни оператори

Всички оператори в частта след `if` и след `else` могат да бъдат произволни оператори, включително и други условни оператори. Тогава се получава вложение на условни оператори. Това дава възможност да се направи избор между повече от две възможности. Тъй като всеки от условните оператори може да бъде в пълен или непълен формат, понякога се създават условия за неоднозначно тълкуване.

Оператор на многопосочно разклонение switch case default

В програмирането често се срещат ситуации изискващи многократно повторение на условен оператор. Много от тези случаи могат да се решат по-бързо и елегантно с оператора `switch` вместо използването на многократно вложени `if else` оператори. Ето един такъв пример, реализиран в двата варианта:

- С помощта на условен оператор

Програмен код

```
#include <stdio.h>
#include <stdlib.h>
#include <conio.h>
#include <ctype.h>
void do_main_menu(short *done);
int main()
{
     short done;
     do_main_menu(&done);
     return 0;
}
void do_main_menu( short *done) {
  char cmd;
  *done = 0;

    printf("\n\t\t**********************************");
    printf("\n\t\t**   (F)ile menu                **");
    printf("\n\t\t**   (R)un                      **");
    printf("\n\t\t**   (C)ompile                  **");
    printf("\n\t\t**   (M)ake                     **");
    printf("\n\t\t**   (P)roject                  **");
    printf("\n\t\t**   (O)ption                   **");
    printf("\n\t\t**   (E)rror                    **");
    printf("\n\t\t**   (Q)uilt                    **");
    printf("\n\t\t**                              **");
    printf("\n\t\t**   Choise symbol              **");
    printf("\n\t\t**********************************");
  do {
    cmd = toupper( (char)getch());
    if (cmd == 'F')       do_main_menu( done);
    else  if (cmd  == 'R') printf("\n\t\t -- run_progràm()      --\n");
    else  if (cmd  == 'C') printf("\n\t\t -- do_compile()      --\n");
    else  if (cmd  == 'M') printf("\n\t\t -- do_make()         --\n");
    else  if (cmd  == 'P') printf("\n\t\t -- do_project()      --\n");
    else  if (cmd  == 'O') printf("\n\t\t -- do_options()      --\n");
    else  if (cmd  == 'E') printf("\n\t\t -- do_errors()       --\n");
    else  if (cmd  == 'Q') *done = 1;
    else                   printf("\n\t\t -- hàndle_others(cmd, done)");
  } while (!*done);
}
```

- с помощта на оператор **switch:**

Програмен код

```
#include <stdio.h>
#include <stdlib.h>
#include <conio.h>
#include <ctype.h>
void do_main_menu(short *done);
int main()
{
     short done;
     do_main_menu(&done);
     return 0;
}
void do_main_menu( short *done) {
  char cmd;
  *done = 0;

    printf("\n\t\t**********************************");
    printf("\n\t\t**    (F)ile menu                **");
    printf("\n\t\t**    (R)un                      **");
    printf("\n\t\t**    (C)ompile                  **");
    printf("\n\t\t**    (M)ake                     **");
    printf("\n\t\t**    (P)roject                  **");
    printf("\n\t\t**    (O)ption                   **");
    printf("\n\t\t**    (E)rror                    **");
    printf("\n\t\t**    (Q)uilt                    **");
    printf("\n\t\t**                               **");
    printf("\n\t\t**    Choise symbol              **");
    printf("\n\t\t**********************************");
  do {
   cmd = toupper( (char)getch());
   switch(cmd) {
    case  'F' :       do_main_menu( done);                    break;
    case  'R' : printf("\n\t\t -- run_program()      --\n"); break;
    case  'C' : printf("\n\t\t -- do_compile()      --\n"); break;
    case  'M' : printf("\n\t\t -- do_make()         --\n"); break;
    case  'P' : printf("\n\t\t -- do_project()      --\n"); break;
    case  'O' : printf("\n\t\t -- do_options()      --\n"); break;
    case  'E' : printf("\n\t\t -- do_errors()       --\n"); break;
    case  'Q' : *done = 1;                                  break;
    default   : printf("\n\t\t -- hàndle_others(cmd,done)"); break;
   }
  } while (!*done);
}
```

Условен израз (?:)

Има случаи, в които на базата на определено условие избираме между два израза (респективно техните стойности). Това действие обикновено се извършва с помощта на **if...else.**

Например:

```
int imin(int a, int b);
void main()
{
   printf("По-малкото число е %d",imin(5,3));
}
int imin(int a, int b)
{
   if (a<b) return (a);
   else return (b);
}
```

Ситуацията се среща много често и затова C++ предвижда специална конструкция, осигуряваща съкратен запис. Форматът е:

Израз1 ? Израз2 : Израз3

Той се интерпретира така: Ако **Израз1** има стойност "истина", да се изпълни **Израз2** и целият израз да приеме неговата стойност. В противен случай да се изпълни **Израз3** и стойността му да стане стойност на целия израз. Следователно горният пример може да се запише по следния начин:

```
int imin(int a, int b);
void main()
 {
  printf("\n\t\a По-малкото число е %d",imin(5,3));
 }

int imin(int a, int b){
  return ((a<b) ? a : b);
 }

float imin(float a, float b);

void main()
{
 printf("\n\t\a По-малкото число е %f",imin(5.8,3.7));
}

float imin(float a, float b)
 {
  return ((a<b) ? a : b);
 }
```

Така, щом програмата срещне израза **imin(e1,e2)**, тя го замества с `((e1<e2) ? e1 : e2)` и компилацията й продължава. Това действително е по-добро и общо решение, тъй като вече не е задължително `e1` и `e2` да са от тип `int`. Те могат да бъдат от произволен тип, който позволява прилагането на операцията `"<"`.

IV. Задание за работа

Зад. 1.. Напишете програма която да извършва следните операции:
- Въвеждане на две цели числа от клавиатурата.
- Сравняване на числата и се извежда по-малкото от двете.
- Извеждане на резултата в дясно подравнен формат
- Извеждане на резултата в шестнадесетичен вид.
- Въвеждане на две реални числа. Да се изведе по-голямото от двете числа.

Зад. 2.. Напишете програма която да извършва следните операции:
- да се въведе едно цяло число
- да се определи дали е четно или нечетно

Зад. 3.Напишете програма която да извършва следните операции:
- да се въведат две цели числа

- да се разделят двете числа
- да се определи остатъка и от целочисленото им деление
- ако резултатът от делението е точно число да се изведе на екрана. В противен случай да се изведе съобщението "Двете числа не се делят точно"

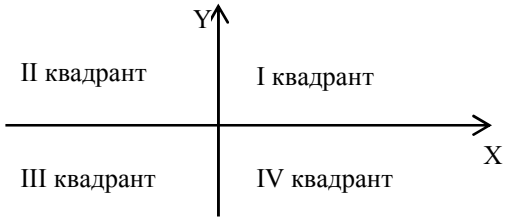
Зад. 4. Напишете програма която да извършва следните операции:
- при въвеждане на четно число да бибитка 2 пъти
- при въвеждане на нечетно число да бибитка 3 пъти.
- Да определи остатъка от делението на -. При остатък по-голям от 3 да сумира числото и остатъка. В противен случай от числото да се извади остатъка.

Зад. 5. Напишете програма която да извършва следните операции:
- да решава квадратно и биквадратно уравнение
- параметрите на уравнението да се въвеждат в процеса на работа

Зад. 6 Да се напише програма, която въвежда стойности за три числа (*x, y* и *z*) и извежда на екрана стойността на най-голямото от тях.

Зад. 7. Три отсечки образуват триъгълник тогава и само тогава, когато най-голямата от тях е по-малка от сумата на другите две (c < a + b). Да се напише програма, която въвежда стойности за трите отсечки и определя дали могат да образуват триъгълник.

Зад. 8 Като се знае, че една точка в равнината се описва с нейните координати (*x* и *y*), да се напише програма, която определя и извежда на екрана в кой квадрант се намира въведена от потребителя точка.



Зад. 9. Да се напише програма, която въвежда цяло число от 1 до 12 и извежда името на съответния месец. В случай, че е въведено число извън този диапазон, програмата да извежда съобщение за грешка.

Зад. 10 Да се напише програма, която изпълнява действията събиране, изваждане, умножение и деление на цели числа.

Зад. 11. Да се напише програма, която представя на екрана следното меню:

1. България
2. Русия
3. Германия
4. Франция
5. Италия

При избор на номер от менюто да се извежда столицата на съответната държава. При неправилен избор да се издава съобщение

за грешка.

V. Указания за работа

При изпълнението на задачите използвайте примерите приложени в теоретичната част?

VI. Контролни въпроси

1. Кои операции се наричат логически?
2. Какви логически оператори познавате?
3. Направете сравнение между оператора if else и switch
4. Направете сравнение между оператора if else и ? :
5. Направете сравнение между оператора ? : и switch
6. Кои програми се наричат линейни?
7. Кои програми се наричат разклонен?
8. Каква стойност за **b** ще се изведе на екрана при изпълнението на следните редове, ако за **a** е въведена стойност : a) 10; b) 0; c) -10?

```
b=0;
if (a>0) b=a+20;
cout<<"b="<<b;
```

9. Каква стойност за **b** ще се изведе на екрана при изпълнението на следните редове, ако за **a** е въведена стойност : a) 10; b) 0; c) -10?

```
b=0;
if (a>0) b=a+20;
else b=a-20;
cout<<"b="<<b;
```

10. Какви стойности за **a** и **b** ще се изведат на екрана при изпълнението на следните редове, ако за **a** е въведена стойност : a) 10; b) 0; c) -10?

```
b=0;
if (a>0) a=a+20;
         b=a-3;
cout<<"b="<<b<<"\na="<<a;
```

11. Какви стойности за **a** и **b** ще се изведат на екрана при изпълнението на следните редове, ако за **a** е въведена стойност : a) 10; b) 0; c) -10?

```
b=0;
if (a>0)
{
  a=a+20;
  b=a-3;
}
cout<<"b="<<b<<"\na="<<a;
```

12 Какви стойности за **a** и **b** от тип int ще се изведат на екрана при изпълнението на следните редове, ако за **a** и **b** се въведат следните стойности :

a) **a** = 5, **b** = 3;
b) **a** = -5, **b** = 3?

```
if (a>0)
{ a=a+20;
  b=a/3;
}
else b=a-20;
cout<<"b="<<b<<"\na="<<a;
```

13. Каква стойност за **b** ще се изведе на екрана при изпълнението на следните редове, ако за **a** и **b** се въведат следните стойности :

a) **a** = 5, **b** = 3;
b) **a** = 5, **b** = -3;
c) **a** = -5, **b** = 3;
d) **a** = -5, **b** = -3?

```
if (a>0) if (b>0) b=a+20;
           else b=a-20;
cout<<"b="<<b;
```

14. Каква стойност за **s** ще се изведе на екрана при изпълнението на следните редове , ако за **n** , **m** и **x** се въведат следните стойности :

a) **n** = 5, **m** = 2.5, **x** = 1;
b) **n** = 1, **m** = 2.5, **x** = 5;
c) **n** = 5.5, **m** = 1, **x** = 3;
d) **n** = 5, **m** = 2.5, **x** = 2?

```
switch (x)
{
  case 1: s=(n+ m)/5;
  case 2: s=n+m/5;
  case 3: s=(n-m)/5;
  default: s=0;
}
cout<<"s="<<s;
```

15 Каква стойност за **s** ще се изведе на екрана при изпълнението на следните редове , ако за **n** , **m** и **x** се въведат следните стойности :

a) **n** = 5, **m** = 2.5, **x** = 1;
b) **n** = 1, **m** = 2.5, **x** = 5;
c) **n** = 5.5, **m** = 1, **x** = 3;
d) **n** = 5, **m** = 2.5, **x** = 2?

```
switch (x)
{
  case 1: s=(n+m)/5;break;
  case 2: s=n+m/5; break;
```

```
        case 3: s=(n-m)/5; break;
        default: s=0;
      }
      cout<<"s="<<s;
```

16. Каква стойност за Y ще се изведе при изпълнението на алгоритъма на фиг.2 за X = **5** и Y = **-2** ?

a) -5
b) 15
c) 50
d) -20

VII. Литература

1. Кай Хорстман. Принципи на програмирането със C++. ИК Софтех София 2000
2. Симеонов Г. Програмиране на C++. С. Техника 1993г.
3. Тед Фейсон. Borland C++. Обектно-ориентирано програмиране. NISOFT София 1994
4. Богданов Д., И.Мустакеров. Език за програмиране С. С. Техника 1989г.
5. А.Касткин. Профессиональное программирование на языке С. Системное программирование. Минск Вышэйшая школа 1993.