Практическая работа №2

Составление и отладка разветвляющихся алгоритмов

1 Цель работы

1.1 Научиться составлять простейшие алгоритмы, используя основные алгоритмические конструкции;

1.2 Научиться составлять полные и неизбыточные наборы тестов.

2 Литература

2.1 Ашарина, И.В. Объектно-ориентированное программирование в C++: лекции и упражнения: учебное пособие для вузов. - Москва: Горячая линия - Телеком, 2014.

2.3 Голицына, О.Л. Программирование на языках высокого уровня: учебное пособие для студентов учреждений среднего проф. образования. - Москва: ФОРУМ, 2018.

3 Основное оборудование

3.1 Персональный компьютер.

4 Подготовка к работе

4.1 Повторить основные алгоритмические конструкции, принципы алгоритмизации;

4.2 Подготовить бланк отчета.

5 Задание

5.1 Написать код для предложенных заданий из п. 6.1-6.4

5.2 Составить отчет в электронном виде с помощью LibreOffice Writer.

5.3 Сохранить работу по пути C:\Temp\КСК-31\Практическая работа 1

6 Порядок выполнения работы

6.1 Пользователь вводит два числа, найти большее из них.

6.2 Цифры 1, 2, 3 и 4 обозначают операции сложение, умножение, вычитание и деление. Предложить пользователю ввести два числа и выбрать действие. Выполнить операцию и вывести результат на экран.

6.3 Исправить ошибку деления на ноль в предыдущем задании.

6.4 Пользователь вводит возраст, если возраст >= 60 выводить «Пенсия одобрена», иначе – «Придется идти на работу». Использовать тернарный оператор!!

7 Содержание отчета

7.1 Титульный лист;

7.2 Цель работы;

7.3 Текст программ (скриншоты);

7.4 Ответы на контрольные вопросы;

7.5 Вывод по проделанной работе.

8 Контрольные вопросы

8.1 Что такое C++ и где используется?

8.2 Как создать новый проект в VisualStudio?

8.3 Какие базовые арифметические операции есть в C++?

.4 Как подключить библиотеку?

9 Приложение

Условная конструкция **if-else** направляет ход программы по одному из возможных путей в зависимости от условия. Она проверяет истинность условия, и если оно истинно, выполняет блок инструкций. В простейшем виде конструкция if имеет следующую сокращенную форму:

```
if (условие)
{
    инструкции;
}
```

В качестве *условия* использоваться условное выражение, которое возвращает **true** или **false**. Если условие возвращает **true**, то выполняются последующие инструкции, которые входят в блок if. Если условие возвращает **false**, то последующие инструкции не выполняются. Блок инструкций заключается в фигурные скобки. Например:

```
#include <iostream>

int main()
{
    int a = 8;
    if(a == 8)
    {
        cout << "a == 8" << endl;
    }
    cout << "End of program" <<  endl;
}
```

Здесь условие конструкции **if** представляет выражение a == 8, то есть мы сравниваем, равно ли значение переменной a числу 8. И это условие верно и возвращает true. Соответственно будет выполняться единственная инструкция из блока if, которая выведет на консоль строку "a == 8". А консольный вывод будет следующим:

```
a == 8
End of program
```

if..else

Также мы можем использовать полную форму конструкции if, которая включает оператор else:

```
if(выражение_условия)
{
    инструкция_1
}
else
{
    инструкция_2
}
```

После оператора else мы можем определить набор инструкций, которые выполняются, если условие в операторе if возвращает false. То есть если условие истинно, выполняются инструкции после оператора if, а если это выражение ложно, то выполняются инструкции после оператора else.

```
#include <iostream>

int main()
{
    int n = 21;
    if (n > 22)
    {
        cout << "n > 22" << endl;
    }
    else
    {
        cout << "n <= 22" << endl;
    }
}
```

В данном случае условие n > 22 ложно, то есть возвращает false, поэтому будет выполняться блок else. И в итоге на консоль будет выведена строка "n <= 22".

Однако нередко надо обработать не два возможных альтернативных варианта, а гораздо больше. Например, в случае выше можно насчитать три

условия: переменная n может быть больше 22, меньше 22 и равна 22. Для проверки альтернативных условий мы можем вводить выражения **else if**:

```
#include <iostream>

int main()
{
    int n = 21;
    if (n > 22)
    {
        cout << "n > 22" << endl;
    }
    else if (n < 22)
    {
        cout << "n < 22" << endl;
    }
    else
    {
        cout << "n == 22" << endl;
    }
}
```

То есть в данном случае мы получаем три ветки развития событий в программе.

Подобных альтернативных условий с помощью выражения else if можно вводить больше одного:

```
#include <iostream>

int main()
{
    int n = 21;
    if (n == 20)
    {
        cout << "n == 20" << endl;
    }
    else if(n==21)
    {
        cout << "n == 21" << endl;
```

```
    }
    else if(n==22)
    {
        cout << "n == 22" << endl;
    }
    else if(n==23)
    {
        cout << "n == 23" << endl;
    }
}
```

Тернарный оператор

Тернарный оператор в некотором роде похож на конструкцию if-else. Он принимает три операнда в следующем виде:

```
операнд1? операнд2 : операнд3
```

Первый операнд представляет условие. Если это условие верно (равно true), тогда выбирается/выполняется второй операнд, который помещается после символа **?**. Если условие не верно, тогда выбирается/выполняется третий операнд, который помещается после двоеточия.

Например, возьмем следующую конструкцию if-else:

```
#include <iostream>

int main()
{
    int a = 5;
    int b =8;
    int c = 0;
    if(a > b)
    {
        c = a - b;
    }
    else
    {
```

```
        c = a + b;
    }
    cout << "c = " << c << endl;  // c = 13
```

Здесь если a больше b, то c=a-b, иначе c=a+b. Перепишем ее с помощью тернарного оператора:

```
#include <iostream>

int main()
{
    int a = 5;
    int b = 8;
    int c = a > b ? a - b : a + b;

    cout << "c = " << c << endl;  // c = 13
}
```

Здесь первым операндом тернарного оператора является условие a > b. Если это условие верно, то возвращается второй операнд - результат выражения a - b. Если условие не верно, то возвращается третий операнд - a + b. И возвращенный операнд присваивается переменной c.

Тернарный оператор не обязательно должен возвращать некоторое значение, он может просто выполнять некоторые действия. Например:

```
#include <iostream>
using namespace std;
int main()
{
    int a = 5;
    int b = 8;
    a > b ? cout << a-b : cout << a+b;
}
```