Линейные алгоритмы. Ветвление. Решение задач.

1) Теория
2) Задачи для самостоятельного решения.
   Для решения задач необходимо зарегистрироваться на сайте
   https://informatics.msk.ru/

Логический тип данных и логические операции

Переменная логического типа может принимать только два значения: *true* (истина) или *false* (ложь). Логический тип данных в языке C++ обозначается словом *bool*.

Например, сохраним в переменную ***b*** результат вычисления выражения $x < 3$:

```
bool b = x < 3;
```

Переменная ***b*** примет значение *true* (истина), если $x < 3$, и *false* (ложь) — в противном случае (если $x \geqslant 3$).

Есть шесть операций сравнения:

- < меньше
- > больше
- <= меньше или равно
- >= больше или равно
- == равно
- != не равно

Приоритет операций сравнения меньше, чем у арифметических операций. Например, если написать $x + 5 < x * 2$, то сначала будет посчитан результат выражений $x + 5$ и $x * 2$ и только затем произойдёт сравнение.

Операции <, >, <=, >= имеют одинаковый приоритет. Операции == и != тоже имеют одинаковый приоритет, но более низкий, чем <, >, <= и >=.

Существуют ещё две логические операции:

- **&&** «логическое И» используется в ситуации, когда должны быть верны оба выражения
- || «логическое ИЛИ» используется в ситуации, когда должно быть верно одно из выражений

Приоритет этих логических операций самый низкий, они выполняются после арифметических операций и операций сравнения. При этом сначала выполняются операции отрицания, потом операции «логического И», и только затем операции «логического ИЛИ». Изменить приоритет выполнения операций можно с помощью круглых скобок.

Проверим, является ли число ***x*** номером месяца:

```
bool c = (x >= 1 && x <= 12);
```

Двойные неравенства в C++ в подобных случаях использовать **нельзя!** Если записать выражение

1 <= *x* <= 12, то сначала выполнится первое сравнение, а затем его результат *true* или *false*, приведённый к числовому значению 1 или 0 соответственно, будет сравниваться с числом 12, в результате всегда получится значение *true*.

Пример 1.

Пусть на дереве было ***n*** белочек и ***x*** орешков, которые они делят между собой поровну (остаток скидывается с дерева). Необходимо проверить, верно ли, что каждой белочке достанется ровно ***y*** орешков.

Проверим также, что задача корректная, а именно количество белок — положительное число.

```
bool ok = n > 0 && x / n == y;
```

Важно то, что если $n = 0$, то первая часть выражения вернёт ложь, поэтому вторая часть выражения вычисляться не будет, а значит не будет деления на 0. Если поменять части логического выражения местами, то программа выдаст ошибку при $n = 0$.

Логический тип данных можно преобразовать в числовой. В этом случае *true* будет преобразован в 1, а *false* в 0.

Пример 2.

Пусть космонавтом может быть человек, который удовлетворяет двум характеристикам из трёх:

- рост ***h*** **<= 170** см;
- вес ***w*** **<= 65** кг;
- ***IQ*** **>= 150**.

Проверим, можем ли мы взять человека с данным ростом, весом и IQ в космонавты:

```
int n = (h <= 170) + (w <= 65) + (IQ >= 150);
bool goToCosmos = n >= 2;
```

Условный оператор if

Пример 3.

По данному числу ***x*** определить его абсолютную величину (модуль). Программа должна напечатать значение переменной ***x***, если ***x*** **> 0**, или же величину **−*x*** в противном случае.

Решение

```
#include <iostream>
using namespace std;
int main()
{
   int x;
   cin >> x;
```

```
    if (x < 0){
        x = -x;
    }
    cout << x;
    return 0;
}
```

В этой программе используется условный оператор *if* (если).

После слова *if* указывается проверяемое логическое выражение ***x* > 0** в круглых скобках. После этого идёт блок (последовательность) инструкций, который будет выполнен, если значение логического выражения истинно. Блок выделяется фигурными скобками.

Данную задачу можно решить другим способом с помощью инструкции *else* (иначе):

```
#include <iostream>
using namespace std;
int main()
{
    int x;
    cin >> x;
    if (x >= 0){
        cout << x;
    } else{
        cout << -x;
    }
    return 0;
}
```

Блок внутри инструкции *else* будет выполнен только в том случае, если логическое выражение внутри *if* вернуло *false* (ложь). Инструкция *else* всегда относится к какой-либо инструкции *if*.

Пример 4. Задача про високосный год

Дано натуральное число. Требуется определить, является ли год с данным номером високосным. Если год является високосным, то выведите YES, иначе выведите NO. Напомним, что в соответствии с григорианским календарём, год является високосным, если его номер делится на 4, но при этом не делится на 100, или если он кратен 400.

Решение

```
#include <iostream>
using namespace std;
int main()
{
    int year;
    cin >> year;
    if (year % 4 == 0){
```

```
        if (year % 100 == 0){
            if (year % 400 == 0){
                cout << "YES" << endl;
            }
            else {
                cout << "NO" << endl;
            }
        } else {
            cout << "YES" << endl;
        }
    } else {
        cout << "NO" << endl;
    }
    return 0;
}
```

Программа получилась довольно громоздкой. Можно написать решение короче:

```
#include <iostream>
using namespace std;
int main()
{
    int year;
    cin >> year;
    if ((year % 4 == 0 && year % 100 != 0) || year % 400 == 0 ){
        cout << "YES" << endl;
    } else {
        cout << "NO" << endl;
    }
    return 0;
}
```

Пример 5.

Даны два числа, выведите минимальное из них.

Решение

```
#include <iostream>
using namespace std;
int main()
{
    int a, b;
    cin >> a >> b;
    if (b < a){
        a = b;
    }
    cout << a;
    return 0;
```

```
}
```

Пример 6.

Даны три числа, выведите минимальное из них.

Решение

```
#include <iostream>
using namespace std;
int main()
{
   int a, b, c;
   cin >> a >> b >> c;
   if (b < a){
      a = b;
   }
   if (c < a){
      a = c;
   }
   cout << a;
   return 0;
}
```

Пример 7.

Пусть дано число ***x***. Необходимо вывести на экран:

- "One", если ***x*** = 1;
- "Two", если ***x*** = 2;
- "Three", если ***x*** = 3;
- "Other" в любом другом случае.

Решение

```
#include <iostream>
using namespace std;
int main()
{
   int x;
   cin >> x;
   if (x == 1){
      cout << "One" << endl;
   }
   if (x == 2){
      cout << "Two" << endl;
   }
   if (x == 3){
      cout << "Three" << endl;
   }
   if (x < 1 || x > 3){
```

```
        cout << "Other" << endl;
    }
    return 0;
}
```

Если понадобится добавить отдельный вывод "Four" для x = 4, то в программу придётся добавить ещё одно условие, а также исправить условие для вывода "Other". Такой подход может привести к ошибкам. Поэтому лучше запись программу иначе:

```
#include <iostream>
using namespace std;
int main()
{
    int x;
    cin >> x;
    if (x == 1){
        cout << "One" << endl;
    } else if (x == 2){
        cout << "Two" << endl;
    } else if (x == 3){
        cout << "Three" << endl;
    } else {
        cout << "Other" << endl;
    }
    return 0;
}
```

При таком способе можно легко модифицировать программу при добавлении других чисел.

Задачи для самостоятельного решения

Для решения задач необходимо зарегистрироваться на сайте https://informatics.msk.ru/

Задача № 560. Змей Горыныч

Задача № 511. Метро

Задача № 564. Кони

Задача № 1416. Шкаф

Задача № 234. День святого Франциска Ксавьера

Задача № 236. Бесконечная таблица

Задача № 304. Билеты на метро

Задача № 525. Яша плавает в бассейне

Задача № 481. Автобусы

Задача №264. Мороженое

Постарайтесь решить как можно больше задач!