

**ТЕМА: ВВЕДЕНИЕ В ЯЗЫК  
ПРОГРАММИРОВАНИЯ «C++»****Домашнее задание 2****ЗАДАНИЕ 1**

Подсчитать количество целых чисел в диапазоне от 100 до 999, у которых есть две одинаковые цифры.

**Подсказка 1**

Если у Вас возникли сложности с выполнением данного задания, нажмите кнопку «Подсказка». С полным решением задания вы сможете ознакомиться, нажав на кнопку «Решение»

**Решение 1****ЗАДАНИЕ 2**

Подсчитать количество целых чисел в диапазоне от 100 до 999, у которых все цифры разные.

**Подсказка 2**

Если у Вас возникли сложности с выполнением данного задания, нажмите кнопку «Подсказка». С полным решением задания вы сможете ознакомиться, нажав на кнопку «Решение»

**Решение 2****ЗАДАНИЕ 3**

Пользователь вводит любое целое число. Необходимо из этого целого числа удалить все цифры 3 и 6 и вывести обратно на экран.

**Подсказка 3**

Если у Вас возникли сложности с выполнением данного задания, нажмите кнопку «Подсказка». С полным решением задания вы сможете ознакомиться, нажав на кнопку «Решение»

**Решение 3**

**ЗАДАНИЕ 4**

Пользователь вводит любое целое число  $A$ . Необходимо вывести все целые числа  $B$ , для которых  $A$  делиться без остатка на  $B \cdot B$  и не делиться без остатка на  $B \cdot B \cdot B$ .

**Подсказка 4**

Если у Вас возникли сложности с выполнением данного задания, нажмите кнопку «Подсказка». С полным решением задания вы сможете ознакомиться, нажав на кнопку «Решение»

**Решение 4****ЗАДАНИЕ 5**

Пользователь вводит целое число  $A$ . Программа должна определить, что куб суммы цифр этого числа равен  $A \cdot A$ .

**ЗАДАНИЕ 6**

Пользователь вводит целое число. Необходимо вывести все целые числа, на которое заданное число делиться без остатка.

**ЗАДАНИЕ 7**

Пользователь вводит два целых числа. Необходимо вывести все целые числа, на которые оба введенных числа делятся без остатка.

Финальные задания рассчитаны на самостоятельное решение. Здесь не будет подсказок и готового алгоритма. Вам необходимо применить все практические навыки, полученные из предыдущих заданий.

## ПОДСКАЗКА К ЗАДАНИЮ 1

1. Какой тип цикла лучше использовать, когда известен диапазон изменений переменной цикла (от ... до ....)?
2. Для определения количества целых чисел, у которых есть две одинаковые цифры, необходимо предварительно разделить число на составляющие его цифры.
3. Любое трехзначное число (число, состоящее из 3 разрядов) можно представить как сумму его цифр, умноженных на 10 в степени, которая соответствует разряду (позиции) цифры в числе справа налево, начиная с нулевой позиции). Например, число  $238 = 2 \cdot 10^2 + 3 \cdot 10^1 + 8 \cdot 10^0$ .
4. Выделение цифры из числа последовательно (столько раз сколько цифр в числе) можно выполнять также справа налево, т. е. от 0-вой позиции до последней (второй, т. к. номера позиций начинаются с 0).
5. Для выделения последней цифры на каждом шаге (на первом шаге это все число, на втором — оставшиеся первые 2 цифры) можно использовать операцию получения остатка от деления числа на 10.
6. Для выделения оставшейся части числа можно использовать операцию целочисленного деления числа на 10.
7. Трехзначное число содержит две одинаковые цифры, если, например, первая цифра равна второй, но при этом не равна третьей, и вторая цифра также не равна третьей.
8. Счетчик количества одинаковых цифр необходимо обнулять для каждого нового анализируемого числа.

## ПОДСКАЗКА К ЗАДАНИЮ 2

1. Какой тип цикла лучше использовать, когда известен диапазон изменений переменной цикла (от ... до ...)?
2. Для определения количества целых чисел, у которых есть две одинаковые цифры, необходимо предварительно разделить число на составляющие его цифры.
3. Любое трехзначное число (число, состоящее из 3 разрядов) можно представить как сумму его цифр, умноженных на 10 в степени, которая соответствует разряду (позиции цифры в числе справа налево, начиная с нулевой позиции). Например, число  $238 = 2 \cdot 10^2 + 3 \cdot 10^1 + 8 \cdot 10^0$ .
4. Выделение цифры из числа последовательно (столько раз сколько цифр в числе) можно выполнять также справа налево, т. е. от нулевой позиции до последней (второй, т. к. номера позиций начинаются с 0).
5. Для выделения последней цифры на каждом шаге (на первом шаге это все число, на втором — оставшиеся первые 2 цифры) можно использовать операцию получения остатка от деления числа на 10.
6. Для выделения оставшейся части числа можно использовать операцию целочисленного деления числа на 10.
7. Трехзначное число не содержит одинаковых цифр, если, первая цифра не равна второй и не равна третьей, и вторая цифра также не равна третьей.

## ПОДСКАЗКА К ЗАДАНИЮ 3

1. Необходимо проанализировать каждую цифру, выделяя ее из числа одну за другой.
2. Выделение цифры из числа можно выполнять справа налево, отделяя и анализируя последнюю цифру вначале полного числа (состоящего из  $N$  цифр), затем — последнюю цифру оставшейся части числа (первые  $N-1$  цифр) и т. д.
3. Для выделения последней цифры на каждом шаге можно использовать операцию получения остатка от деления текущего числа на 10.
4. Для выделения оставшейся части числа (которую необходимо обрабатывать далее) можно использовать операцию целочисленного деления числа на 10.
5. В процессе извлечения цифр число уменьшается, значит данный процесс повторяется, пока число больше нуля.
6. Любое  $N$  — значное число (число, состоящее из  $N$  разрядов) можно представить как сумму его цифр, умноженных на 10 в степени, которая соответствует разряду (позиции цифры в числе справа налево, начиная с нулевой позиции). Например, трехзначное число  $238 = 2 \cdot 10^2 + 3 \cdot 10^1 + 8 \cdot 10^0$ .
7. Если необходимо удалить цифру из числа, то в процессе накопления суммы слагаемое, соответствующее данному числу, пропускается, а его разряд переходит к следующему слева. Например,  $238 = 2 \cdot 10^2 + 3 \cdot 10^1 + 8 \cdot 10^0$  — после удаления цифры 3 — получаем  $28 = 2 \cdot 10^1 + 8 \cdot 10^0$ .

## ПОДСКАЗКА К ЗАДАНИЮ 4

1. Так как число  $A$  должно без остатка делиться, как минимум, на  $B \cdot B$ , то поиск чисел  $B$  следует выполнять в диапазоне  $[0; A)$ , где число ноль будет входить в диапазон, а число  $A$  — нет, проверяя каждое из чисел диапазона на соответствие указанным условиям.
2. Какой тип цикла следует использовать, если известен диапазон значений переменной — счетчика цикла?
3. На каждом шаге цикла анализируемое число  $B$  изменяется, уменьшаясь от  $A-1$  до  $0$  включительно.
4. Если значение переменной — счетчика цикла изменяется от большего к меньшему, то какой тип цикла нужно использовать: прямой или обратный?
5. Какой логический оператор необходимо использовать для проверки на соответствие двум условиям одновременно.

**РЕШЕНИЕ ЗАДАНИЯ 1****Описание решения**

Необходимо организовать анализ каждого числа из указанного диапазона в цикле.

Для перехода к новому числу, счетчик цикла, начальное значения которого совпадает с началом диапазона, необходимо увеличивать на каждом шаге на единицу, т. к. анализируются целые числа.

Перед анализом каждого числа счетчик количества одинаковых цифр необходимо обнулять.

Рассмотрим процесс выделения цифр из числа 123.

На первом шаге к числу применяем операцию остатка от деления на 10, результат — последняя (третья) цифра:

$$123 \% 10 = 3.$$

Для получения оставшейся (для дальнейшей обработки) части числа (первых двух цифр) выполняем операцию целочисленного деления на 10:

$$123 / 10 = 12.$$

Повторяем первый шаг над числом 12 и получаем предпоследнюю (вторую) цифру:

$$12 \% 10 = 2.$$

Для получения первой цифры повторно выполняем над числом 12 операцию целочисленного деления на 10:

$$12 / 10 = 1.$$

Далее организуем последовательность проверок на наличие двух одинаковых цифр, которые возможны в следующих ситуациях (для трехзначного числа):

- если первая цифра равна второй, но при этом первая цифра не равна третьей и вторая цифра не равна третьей (например, 110);
- если первая цифра равна третьей, но при этом первая цифра не равна второй и вторая цифра не равна третьей (например, 101);
- если вторая цифра равна третьей, но при этом первая цифра не равна второй и первая цифра не равна третьей (например, 211).

При срабатывании указанных ситуаций увеличиваем счетчик одинаковых цифр для текущего числа на единицу.

После выполнения всех проверок, если счетчик одинаковых цифр для текущего числа равен единице (т.е. в числе только две одинаковые цифры), то увеличиваем счетчик чисел, у которых есть две одинаковые цифры.

### Решение

1. Выводим строку в консоль с описанием названия программы (по желанию, данная строка кода может быть пропущена), создаем шесть переменных целочисленного типа для хранения трех цифр числа, количества одинаковых цифр в числе, общего количества чисел, у которых есть две одинаковые цифры (результата) и временную для хранения части числа при выделении из него цифр.

Устанавливаем начальное значение переменной результата в ноль.

```
#include <iostream>
using namespace std;
```



```
int main()
{
    cout << "Home task #6.2.1\n\n";
    int number1, number2, number3, same, resN, temp;
    resN = 0;

    return 0;
}
```

2. Для анализа каждого числа из указанного диапазона будем использовать цикл **for**, так как известно необходимое количество итераций (диапазон изменения переменной — счетчика от 100 до 999).

Для перехода к новому числу, счетчик цикла, начальное значения которого совпадает с началом диапазона, необходимо увеличивать на каждом шаге на единицу, т.к. анализируются целые числа.

Перед анализом каждого числа счетчик количества одинаковых цифр необходимо обнулять.

```
#include <iostream>
using namespace std;

int main()
{
    cout << "Home task #6.2.1\n\n";
    int number1, number2, number3, same, resN, temp;
    resN = 0;
    for (int i = 100; i <= 999; i++)
    {
        same = 0;
```

```
    }  
    return 0;  
}
```

3. В теле цикла выполняем разделение каждого числа на составляющие его цифры.

Вначале к числу применяем операцию остатка от деления на 10, результат — последняя (третья) цифра.

Для получения оставшейся (для дальнейшей обработки) части числа (первых двух цифр) выполняем операцию целочисленного деления на 10 и сохраняем полученный результат во временную переменную.

Повторяем операцию целочисленного деления на 10 для временной переменной и получаем предпоследнюю (вторую) цифру. Для получения первой цифры повторно выполняем над временной переменной операцию целочисленного деления на 10.

```
#include <iostream>  
using namespace std;  
  
int main()  
{  
    cout << "Home task #6.2.1\n\n";  
    int number1, number2, number3, same, resN, temp;  
    resN = 0;  
    for (int i = 100; i <= 999; i++)  
    {  
        same = 0;  
        number3 = i % 10;
```

```
temp = i / 10;  
number2 = temp % 10;  
number1 = temp / 10;  
}  
return 0;  
}
```

4. Далее организуем последовательность проверок на наличие двух одинаковых цифр, которые возможны в следующих ситуациях (для трехзначного числа):

- если первая цифра равна второй, но при этом первая цифра не равна третьей и вторая цифра не равна третьей (например, 110);
- если первая цифра равна третьей, но при этом первая цифра не равна второй и вторая цифра не равна третьей (например, 101);
- если вторая цифра равна третьей, но при этом первая цифра не равна второй и первая цифра не равна третьей (например, 211).

При срабатывании указанных ситуаций увеличиваем счетчик одинаковых цифр для текущего числа на единицу.

```
#include <iostream>  
using namespace std;  
  
int main()  
{  
    cout << "Home task #6.2.1\n\n";  
    int number1, number2, number3, same, resN, temp;
```

```
resN = 0;
for (int i = 100; i <= 999; i++)
{
    same = 0;
    number3 = i % 10;
    temp = i / 10;
    number2 = temp % 10;
    number1 = temp / 10;
    if ((number1 == number2) &&
        (number1 != number3) &&
        (number2 != number3))
    {
        same++;
    }
    if ((number1 != number2) &&
        (number1 == number3) &&
        (number2 != number3))
    {
        same++;
    }
    if ((number1 != number2) &&
        (number1 != number3) &&
        (number2 == number3))
    {
        same++;
    }
}

cout << "#of numbers with 2 identical digits:" << resN;
return 0;
}
```

5. Проверяем, если счетчик одинаковых цифр для текущего числа равен единице (т. е. в числе только две одинаковые цифры), то увеличиваем счетчик чисел, у которых есть две

одинаковые цифры. Выводим в консоль данную цифру. После анализа всех цифр диапазона (окончания цикла) выводим полученный результат в консоль.

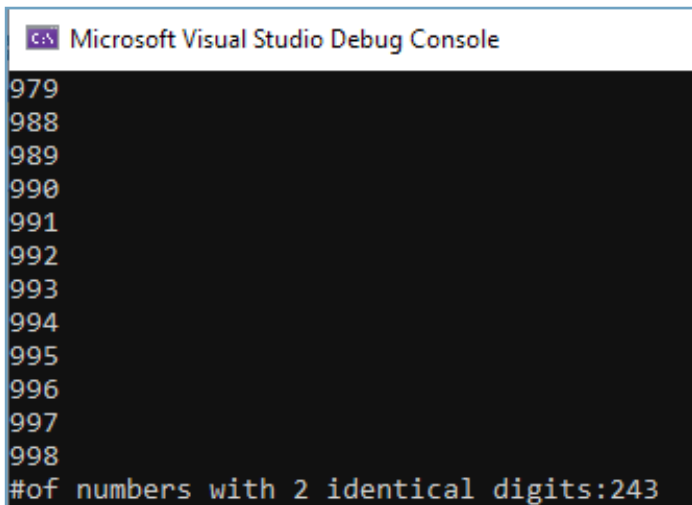
```
#include <iostream>
using namespace std;

int main()
{
    cout << "Home task #6.2.1\n\n";
    int number1, number2, number3, same, resN, temp;
    resN = 0;
    for (int i = 100; i <= 999; i++)
    {
        same = 0;
        number3 = i % 10;
        temp = i / 10;
        number2 = temp % 10;
        number1 = temp / 10;

        if ((number1 == number2) &&
            (number1 != number3) &&
            (number2 != number3))
        {
            same++;
        }
        if ((number1 != number2) &&
            (number1 == number3) &&
            (number2 != number3))
        {
            same++;
        }
        if ((number1 != number2) &&
            (number1 != number3) &&
            (number2 == number3))
```

```
        {  
            same++;  
        }  
        if (same == 1)  
        {  
            cout << i << "\n";  
            resN++;  
        }  
    }  
    cout << "#of numbers with 2 identical digits:"  
        << resN;  
    return 0;  
}
```

Результаты работы программы (в консоли):



```
Microsoft Visual Studio Debug Console  
979  
988  
989  
990  
991  
992  
993  
994  
995  
996  
997  
998  
#of numbers with 2 identical digits:243
```

Рисунок 1

## РЕШЕНИЕ ЗАДАНИЯ 2

## Описание решения

Необходимо организовать анализ каждого числа из указанного диапазона в цикле.

Для перехода к новому числу, счетчик цикла, начальное значения которого совпадает с началом диапазона, необходимо увеличивать на каждом шаге на единицу, т.к. анализируются целые числа.

Перед анализом каждого числа счетчик количества одинаковых цифр необходимо обнулять.

Рассмотрим процесс выделения цифр из числа 123.

На первом шаге к числу применяем операцию остатка от деления на 10, результат — последняя (третья) цифра:

$$123 \% 10 = 3.$$

Для получения оставшейся (для дальнейшей обработки) части числа (первых двух цифр) выполняем операцию целочисленного деления на 10:

$$123 / 10 = 12.$$

Повторяем первый шаг над числом 12 и получаем предпоследнюю (вторую) цифру:

$$12 \% 10 = 2.$$

Для получения первой цифры повторно выполняем над числом 12 операцию целочисленного деления на 10:

$$12 / 10 = 1.$$

Далее организуем проверку, состоит ли данное число из разных цифр. Данная проверка — это одновременное выполнение нескольких (трех) условий:

- если первая цифра не равна второй;
- если первая цифра не равна третьей;
- если вторая цифра не равна третьей;

Например, проверим число 123:

- если первая цифра (1) не равна второй (2) — условие выполняется;
- если первая цифра (1) не равна третьей (3) — условие выполняется;
- если вторая цифра (2) не равна третьей (3) — условие выполняется;

Все три условия выполняются, значит, число 123 состоит из разных цифр.

Например, проверим число 112:

- если первая цифра (1) не равна второй (1) — условие НЕ выполняется;
- если первая цифра (1) не равна третьей (3) — условие выполняется;
- если вторая цифра (2) не равна третьей (3) — условие выполняется;

Первое условие не выполнилось, значит, вся проверка вернула отрицательный результат — число 112 не состоит из разных цифр.

### Решение

1. Выводим строку в консоль с описанием названия программы (по желанию, данная строка кода может быть пропущена), создаем пять переменных целочисленного типа для хранения трех цифр числа, общего количества



чисел, у которых есть все цифры разные (результата) и временную для хранения части числа при выделении из него цифр.

Устанавливаем начальное значение переменной результата в ноль.

```
#include <iostream>
using namespace std;

int main()
{
    cout << "Home task #6.2.2\n\n";
    int number1, number2, number3, resN, temp;
    resN = 0;
    return 0;
}
```

2. Для анализа каждого числа из указанного диапазона будем использовать цикл `for`, так как известно необходимое количество итераций (диапазон изменения переменной — счетчика от 100 до 999).

Для перехода к новому числу, счетчик цикла, начальное значения которого совпадает с началом диапазона, необходимо увеличивать на каждом шаге на единицу, т.к. анализируются целые числа.

```
#include <iostream>
using namespace std;

int main()
{
```

```
cout << "Home task #6.2.2\n\n";
int number1, number2, number3, resN, temp;

resN = 0;
for (int i = 100; i <= 999; i++)
{
}
return 0;
}
```

3. В теле цикла выполняем разделение каждого числа на составляющие его цифры.

Вначале к числу применяем операцию остатка от деления на **10**, результат — последняя (третья) цифра:

Для получения оставшейся (для дальнейшей обработки) части числа (первых двух цифр) выполняем операцию целочисленного деления на **10** и сохраняем полученный результат во временную переменную.

Повторяем операцию целочисленного деления на **10** для временной переменной и получаем предпоследнюю (вторую) цифру.

Для получения первой цифры повторно выполняем над временной переменной операцию целочисленного деления на **10**.

```
#include <iostream>
using namespace std;

int main()
{
    cout << "Home task #6.2.2\n\n";
```

```
int number1, number2, number3, resN, temp;
resN = 0;

for (int i = 100; i <= 999; i++)
{
    same = 0;
    number3 = i % 10;
    temp = i / 10;
    number2 = temp % 10;
    number1 = temp / 10;
}
return 0;
}
```

4. Далее организуем проверку, состоит ли данное число из разных цифр. Данная проверка — это одновременное выполнение нескольких (трех) условий (поэтому будет использоваться логический оператор **И**):

- если первая цифра не равна второй;
- если первая цифра не равна третьей;
- если вторая цифра не равна третьей.

После анализа всех цифр диапазона (окончания цикла) выводим полученный результат в консоль.

```
#include <iostream>
using namespace std;

int main()
{
    cout << "Home task #6.2.2\n\n";
```

```
int number1, number2, number3, same, resN, temp;

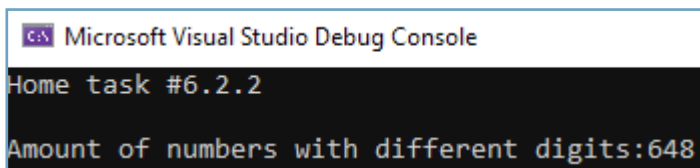
resN = 0;

for (int i = 100; i <= 999; i++)
{
    same = 0;
    number3 = i % 10;
    temp = i / 10;
    number2 = temp % 10;
    number1 = temp / 10;
    if ((number1 != number2) &&
        (number1 != number3) &&
        (number2 != number3))
    {
        resN ++;
    }
}

cout << "Amount of numbers with different digits:";
cout << resN;

return 0;
}
```

Результаты работы программы (в консоли):



Microsoft Visual Studio Debug Console

Home task #6.2.2

Amount of numbers with different digits:648

Рисунок 2

**РЕШЕНИЕ ЗАДАНИЯ 3****Описание решения**

Необходимо организовать процесс извлечения и анализа последней цифры текущего числа в цикле.

На первом шаге текущее число — это оригинальное число, заданное пользователем. Для извлечения последней цифры применяем операцию остатка от деления на 10, результат — последняя цифра.

Рассмотрим процесс на примере числа 238.

На первом шаге к числу применяем операцию остатка от деления на 10, результат — последняя (третья) цифра:

$$238 \% 10 = 8.$$

Следующее (подлежащее дальнейшей обработке) число — это начальная часть текущего числа без отделенной последней цифры.

Для получения оставшейся части числа (первых двух цифр в нашем случае) выполняем операцию целочисленного деления на 10:

$$238 / 10 = 23.$$

Т.е. на следующем этапе (следующий шаг цикла) будет обрабатываться число 23.

Так как нам необходимо не только выделить цифры исходного числа, но и собрать новое число (без цифр 3 и 8, если они входили в состав исходного), то на каждом шаге цикла также нужно собирать это новое число.

Процесс сбора числа — это процесс накопления суммы, каждым слагаемым которой является выделенная цифры (при условии, что она не равна 3 или 8), умноженная на 10 в

степени, которая соответствует разряду (позиции цифры в числе справа налево, начиная с нулевой позиции).

При этом увеличение разряда на единицу (при переходе на следующий шаг цикла) происходит, только, если выделенное число значимое (не 3 и не 6).

Т.к. пользователь может ввести любое  $N$  — значное число, то количество повторений цикла неизвестно. Значит, необходимо использовать тип цикла с предусловием (пока существует число, из которого можно выделять цифры, т.е. пока обрабатываемое число больше 0).

Для нашего примера (число 238):

#### Шаг 1:

- текущее число = 238;
- разряд = 0;
- выделенная цифра  $238 \% 10 = 8$  (не 3 и не 6), значит, накапливаем сумму и разряд:
  - ▶ новое число =  $8 * 100$ ,
  - ▶ разряд = разряд + 1 = 1;
- текущее число =  $238 / 10 = 23$ .

#### Шаг 2:

- текущее число = 23;
- разряд = 1;
- выделенная цифра  $23 \% 10 = 3$ , значит, НЕ накапливаем сумму и разряд;
- текущее число =  $23 / 10 = 2$ .

#### Шаг 3:

- текущее число = 2;
- разряд = 1;

- выделенная цифра  $2 \% 10 = 2$ , значит, накапливаем сумму и разряд:
  - ▶ новое число  $= 2 * 101 + 8 * 100$ ,
  - ▶ разряд  $= \text{разряд} + 1 = 2$ ;
- текущее число  $= 2 / 10 = 0$ .

Конец обработки.

Результат: новое число  $= 2 * 101 + 8 * 100 = 28$ .

### Решение

1. Выводим строку в консоль с описанием названия программы (по желанию, данная строка кода может быть пропущена), создаем четыре переменных целочисленного типа для хранения исходного числа, выделенной цифры, разряда и нового числа (результата).

Выводим в консоль строку — запрос с просьбой ввести целое число.

Считываем введенное пользователем число в соответствующую переменную.

Устанавливаем начальное значение переменной результата и переменной разряда в ноль.

```
#include <iostream>
using namespace std;

int main()
{
    cout << "Home task #6.2.2\n\n";
    int number, digit, i, newNumber;
    cout << "Please, enter your number:\n";
```

```
    cin >> number;
    i = 0;
    newNumber = 0;

    return 0;
}
```

2. Организуем цикл с предусловием: будем продолжать процесс выделения, анализа и формирования нового числа, пока существует обрабатываемое число, из которого можно выделять цифры, т.е. пока обрабатываемое число больше 0. В цикле для извлечения последней цифры применяем над текущим числом операцию остатка от деления на 10, результат — последняя цифра.

```
#include <iostream>
using namespace std;

int main()
{
    cout << "Home task #6.2.2\n\n";
    int number, digit, i, newNumber;

    cout << "Please, enter your number:\n";
    cin >> number;

    i = 0;
    newNumber = 0;

    while (number > 0)
    {
```



```
        digit = number % 10;
    }

    return 0;
}
```

3. Проверяем, является ли выделенная цифра тройкой или шестеркой. Если нет, то накапливаем новое число, добавляя к нему выделенную цифру, умноженную на десять в степени разряда, после чего увеличиваем значения разряда на единицу.

Иначе (выделенная цифра равна трем или шести, не включается в новое число), накопления нового числа и увеличения значения разряда не происходит.

Формируем новое число (подлежащее дальнейшей обработке), выполняя над текущим числом операцию целочисленного деления на 10.

Выводим полученный результат в консоль.

```
#include <iostream>
using namespace std;

int main()
{
    cout << "Home task #6.2.2\n\n";
    int number, digit, i, newNumber;

    cout << "Please, enter your number:\n";
    cin >> number;
    i = 0;
```

```
newNumber = 0;

while (number > 0)
{
    digit = number % 10;
    if ((digit != 3) && (digit != 6))
    {
        newNumber = newNumber + digit * pow(10, i);
        i++;
    }

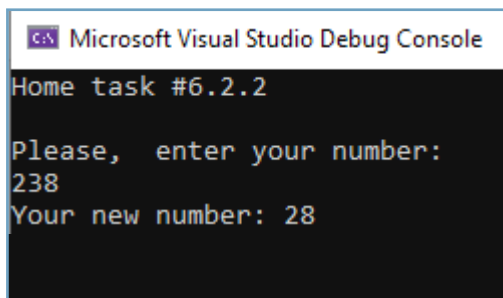
    number = number / 10;
}

cout << "Your new number: " << newNumber;

return 0;
}
```

Результаты работы программы (в консоли):

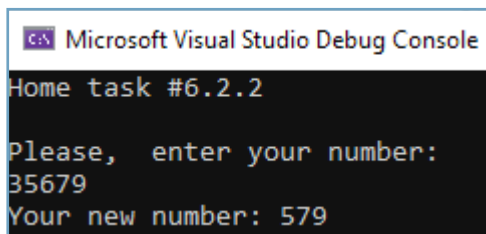
- Тест 1 — введенное трехразрядное число содержит цифру 3.



```
Microsoft Visual Studio Debug Console
Home task #6.2.2
Please, enter your number:
238
Your new number: 28
```

Рисунок 3

- Тест 2 — введенное пятиразрядное число содержит цифры 3 и 6.

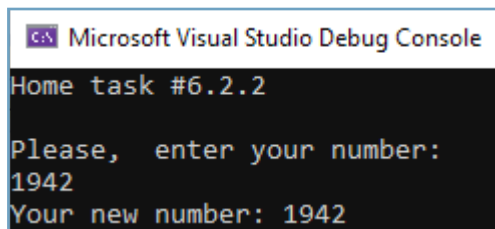


```
Microsoft Visual Studio Debug Console
Home task #6.2.2

Please, enter your number:
35679
Your new number: 579
```

Рисунок 4

- Тест 3 — введенное четырехразрядное число не содержит цифры 3 и 6.



```
Microsoft Visual Studio Debug Console
Home task #6.2.2

Please, enter your number:
1942
Your new number: 1942
```

Рисунок 5

## РЕШЕНИЕ ЗАДАНИЯ 4

## Описание решения

Так как число  $A$  должно без остатка делиться, как минимум, на  $B*B$ , то поиск чисел  $B$  следует выполнять в диапазоне  $[0;A)$ , двигаясь от числа  $A$  до нуля.

Проверку условий «делится / не делиться без остатка» выполняем с помощью оператора  $\%$  — остаток от деления, сравнения результат с нулем:

$X \% Y == 0$  —  $X$  делится без остатка на  $Y$ ;

$X \% Y != 0$  —  $X$  не делится без остатка на  $Y$ .

Число  $A$  не включается в проверку, так число не может делиться на квадрат самого себя без остатка.

Единственное число, которое делится на квадрат самого себя без остатка — это единица, но так как в условии задания присутствует второе условие «не делиться без остатка на  $B*B*B$ », то единица не подходит ( $1\%(1 * 1 * 1)=0$ ).

Так как в ходе анализа чисел диапазона будем двигаться от большего числа ( $A-1$ ) к меньшему ( $0$ ), то организуем обратный цикл, после выполнения каждого шага которого переменная — счетчик цикла (анализируемое число) будет уменьшаться на единицу (т.к. по условию задания нужно искать целые числа, а разница между двумя целыми числами, текущим и следующим, равна единице).

Будем выполнять проверку каждого из чисел диапазона на одновременное соответствие указанным условиям с помощью логического оператора «И».

### Решение

1. Выводим строку в консоль с описанием названия программы (по желанию, данная строка кода может быть пропущена), создаем одну целочисленную переменную для хранения исходного числа **A**.

Выводим в консоль строку — запрос с просьбой ввести целое число.

Считываем введенное пользователем число в соответствующую переменную.

```
#include <iostream>
using namespace std;

int main()
{
    cout << "Home task #6.2.4\n\n";
    int A, B;
    cout << "Please, enter your number:\n";
    cin >> A;
    return 0;
}
```

2. Организуем обратный цикл (переменная счетчик будет изменяться от **A-1** до **0**), после выполнения каждого шага которого переменная — счетчик цикла (анализируемое число) будет уменьшаться на единицу.

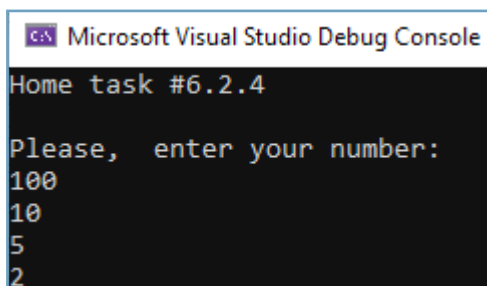
На каждом шаге цикла будем выполнять проверку текущего числа (**B**) на одновременное соответствие двум условиям «**A** делиться без остатка на **B\*B**», «**A** не делиться без остатка на **B\*B\*B**» с помощью логического оператора «**И**».

Если эти два условия одновременно выполняются для текущего числа, то выводим число в консоль.

```
#include <iostream>
using namespace std;

int main()
{
    cout << "Home task #6.2.4\n\n";
    int A, B;
    cout << "Please, enter your number:\n";
    cin >> A;
    for (int i = A-1; i > 1; i--)
    {
        B = i;
        if ((A % (B * B) == 0) && (A % (B * B * B) != 0))
        {
            cout << B << "\n";
        }
    }
    return 0;
}
```

Результаты работы программы (в консоли):



```
Microsoft Visual Studio Debug Console
Home task #6.2.4
Please, enter your number:
100
10
5
2
```

Рисунок 6