

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

Напишите программу, которая вычисляет сумму целых чисел от a до 500 (значение a вводится с клавиатуры).

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**

Напишите программу, которая запрашивает два целых числа x и y , после чего вычисляет и выводит значение x в степени y .

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**

Найти среднее арифметическое всех целых чисел от 1 до 1000.

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**

Найти произведение всех целых чисел от a до 20 (значение a вводится с клавиатуры: $1 \leq a \leq 20$).

ЗАДАНИЕ 5

Написать программу, которая выводит на экран таблицу умножения на k , где k — номер варианта. Например, для 7-го варианта:

$$7 \times 2 = 14;$$

$$7 \times 3 = 21.$$

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. Вычисление суммы чисел в заданном диапазоне происходит путем накопления, т. е. на каждом шаге цикла к текущей сумме добавляется новое (следующее) число из диапазона.
2. Будет использоваться цикл с предусловием или постусловием?
3. Какое будет условие выполнения цикла?
4. Какое действие (действия) будут располагаться в теле цикла?
5. Следует ли помещать в тело цикла операцию вывода суммы в консоль?

ПОДСКАЗКА К ЗАДАНИЮ 2

1. Вычисление степени числа происходит путем последовательного умножения числа на него же такое количество раз, которое соответствует указанной степени.
2. Будет использоваться цикл с предусловием или постусловием?
3. Какое будет условие выполнения цикла?
4. Какое действие (действия) будут располагаться в теле цикла?
5. Следует ли помещать в тело цикла операцию вывода результата в консоль?

ПОДСКАЗКА К ЗАДАНИЮ 3

1. Нахождение среднего арифметического чисел некоторого диапазона основывается на вычисление суммы чисел в заданном диапазоне с последующим делением данной суммы на количество чисел в диапазоне.
2. Вычисление суммы чисел в заданном диапазоне происходит путем накопления, т. е. на каждом шаге цикла к текущей сумме добавляется новое (следующее) число из диапазона.
3. Какой тип цикла лучше использовать, когда известно количество повторений?
4. Какое будет условие выполнения цикла?
5. Нужно ли считать количество элементов (целых чисел) в диапазоне от 1 до 1000?
6. Какое действие будет располагаться в теле цикла?
7. Операцию деления суммы на количество чисел в диапазоне следует помещать в тело цикла или выполнять после него?

РЕШЕНИЕ ЗАДАНИЯ 1

Описание решения

Пусть требуется вычислить сумму:

$$S = a_1 + a_2 + \dots + a_n$$

Решение сводится к последовательному вычислению промежуточных сумм:

$$S_0 = 0$$

начальное значение, устанавливается перед вычислениями;

$$S_1 = S_0 + a_1;$$

$$S_2 = S_1 + a_2;$$

...

$$S_n = S_{n-1} + a_n = a_1 + a_2 + \dots + a_n.$$

Вычисление значения S_n представляет собой искомую сумму S . Значение промежуточных сумм S_1, \dots, S_{n-1} не требуется сохранять, поэтому последовательность вычислений, представленную выше, можно сформулировать в виде общей формулы:

$$S = S + a_i,$$

где a_i – слагаемое на i — том шаге.

Таким образом, вычисление суммы сводится к ее накоплению в переменной S на каждом шаге цикла.

Например, необходимо найти сумму чисел от 1 до 5 включительно.

Наши слагаемые:

$$a_1 = 1;$$

$$a_2 = 2;$$

$$a_3 = 3;$$

$$\begin{aligned}a_4 &= 4; \\ a_5 &= 5; \\ S &= a_1 + a_2 + a_3 + a_4 + a_5; \\ S &= 1 + 2 + 3 + 4 + 5.\end{aligned}$$

Перед началом вычисления суммы еще нет, т.е. ее значение равно нулю.

$$\begin{aligned}S_0 &= 0; \\ S_1 &= S_0 + a_1 = 0 + 1 = 1; \\ S_2 &= S_1 + a_2 = 1 + 2 = 3; \\ S_3 &= S_2 + a_3 = 3 + 3 = 6; \\ S_4 &= S_3 + a_4 = 6 + 4 = 10; \\ S_5 &= S_4 + a_5 = 10 + 5 = 15.\end{aligned}$$

Решение

1. Выводим строку в консоль с описанием названия программы (по желанию, данная строка кода может быть пропущена), создаем необходимое число переменных: две целочисленные для хранения введенного пользователем целого числа a и результата (суммы чисел диапазона). Устанавливаем начальное значение переменной суммы в ноль, так как в начале процесса вычислений суммы еще нет, т.е. она равна нулю. 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 #6.1.1\n\n";  
    int a, sum;  
    sum = 0;  
    cout << "Enter the number:\n";  
    cin >> a;  
  
    return 0;  
}
```

2. Организуем цикл: будем накапливать сумму (продолжать сложение), пока текущее число, двигаясь от начального значения (**a**) не достигнет значение **500**.

Кроме операции по накоплению суммы в теле цикла необходимо организовать продвижение числа на следующую позицию диапазона. Так как по условию складываются целые числа, то для передвижения к следующему целому числу необходимо к текущему числу прибавить один.

После завершения цикла сумма готова и ее значение можно вывести пользователю в консоль.

```
#include <iostream>  
using namespace std;  
  
int main()  
{  
    cout << "Home task #6.1.1\n\n";  
    int a, sum;  
    sum = 0;  
    cout << "Enter the number:\n";
```



```
cin >> a;

while (a <= 500)
{
    sum = sum + a;
    a = a + 1;
}

cout << "Sum = " << sum;
return 0;
}
```

Результаты работы программы (в консоли):

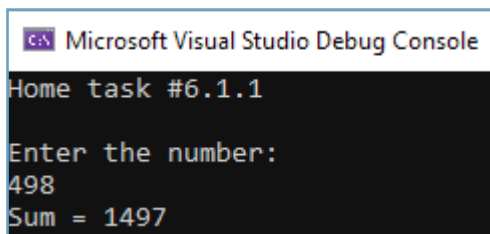


Рисунок 1

РЕШЕНИЕ ЗАДАНИЯ 2

Описание решения

Для того, чтобы возвести число в степень, его надо умножить само на себя количество раз, равное показателю степени. Фактически, решение сводится к последовательному вычислению промежуточных произведений y раз:

$$P_0 = 1,$$

начальное значение, устанавливается перед вычислениями, не 0, т. к. каждое следующее умножение на 0 также даст 0);

$$P_1 = P_0 * x;$$

$$P_2 = P_1 * x;$$

...

$$P_y = P_{y-1} * x = x * x * \dots * x.$$

Вычисление значения P_y представляет собой искомый результат (x^y). Значение промежуточных произведений P_1, \dots, P_{y-1} не требуется сохранять, поэтому последовательность вычислений, представленную выше, можно сформулировать в виде общей формулы:

$$P = P * x, \tag{1}$$

Данное действие (1) необходимо повторить y раз.

Например, нужно вычислить 23:

$$P_0 = 1;$$

$$P_1 = P_0 * x = 1 * 2 = 2;$$

$$P_2 = P_1 * x = 2 * 2 = 4;$$

$$P_3 = P_2 * x = 4 * 2 = 8.$$

Если y равно нулю, то, какое бы число не стояло в основании степени, результат всегда будет равен единице.

Если показатель степени отрицателен ($y < 0$), то результат определяется такой формулой:

$$P = 1 / P. \quad (2)$$

Решение

1. Выводим строку в консоль с описанием названия программы (по желанию, данная строка кода может быть пропущена), создаем необходимое число переменных: четыре целочисленные для хранения введенного пользователем целого числа, его степени, результата и счетчика цикла (какой раз происходит операция умножения).

Устанавливаем начальное значение переменной результата равное единице. Начальное значение, устанавливается перед вычислениями, не 0, т. к. каждое следующее умножение на 0 также даст 0.

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

Считываем введенное пользователем число в соответствующую переменную.

Повторяем запрос для ввода степени числа.

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

int main()
{
    cout << "Home task #6.1.2\n\n";
    int x, y, i;
    float power;
    power = 1;
```

```
cout << "Enter the number X:\n";  
cin >> x;  
cout << "Enter the number Y:\n";  
cin >> y;  
  
return 0;  
}
```

2. Выполняем проверку степени числа (y). Для реализации процедуры сравнения будем использовать оператор условного ветвления **if-else**.

Если y равно нулю, то результат будет равен единице.

Иначе реализуем процесс вычислений, т. к. и для положительной, и для отрицательной степени нам необходимо накопить произведение.

Устанавливаем начальное значение счетчика цикла равное единице (номер шага умножения).

Количество повторений для цикла равно степени числа (y).

Однако, для случая, когда степень числа отрицательная ($y < 0$), цикл не запустится ни разу, т. к. условие работы цикла «пока номер шага умножения меньше или равно y », а начальное значение шага умножения равно 1 (т.е. больше 0).

Поэтому в условии цикла нужно использовать абсолютное значение (модуль) y , т. к. количество повторений не может быть отрицательным числом.

Кроме операции умножения в теле цикла необходимо организовать увеличение счетчика цикла на единицу, например, выполним второе умножение переходим к третьему.

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

int main()
{
    cout << "Home task #6.1.2\n\n";
    int x, y, i;
    float power;

    power = 1;

    cout << "Enter the number X:\n";
    cin >> x;

    cout << "Enter the number Y:\n";
    cin >> y;

    if (y == 0)
    {
        power = 1;
    }
    else
    {
        i = 1;

        while (i <= abs(y))
        {
            power = power * x;
            i = i + 1;
        }

        return 0;
    }
}
```

3. После накопления произведения реализуем проверку, является ли степень числа отрицательной. Если да, то перепределяем результат по формуле (2). Иначе результат не изменяется.

После завершения цикла возведения числа x в степень y готов и его значение можно вывести пользователю в консоль.

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

int main()
{
    cout << "Home task #6.1.2\n\n";
    int x, y, i;
    float power;

    power = 1;

    cout << "Enter the number X:\n";
    cin >> x;

    cout << "Enter the number Y:\n";
    cin >> y;

    if (y == 0)
    {
        power = 1;
    }
    else
    {
        i = 1;
```

```
while (i <= abs(y))
{
    power = power * x;
    i = i + 1;
}

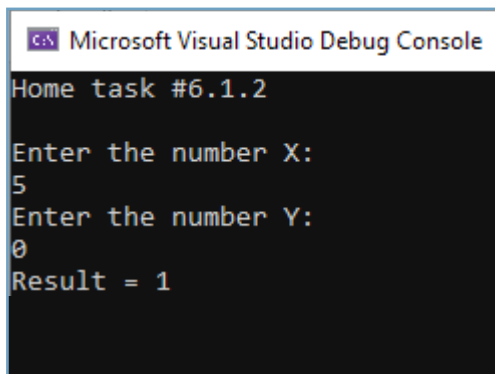
if (y < 0)
{
    power = 1 / power;
}

cout << "Result = " << power;

return 0;
}
```

Результаты работы программы (в консоли):

- Тест 1 — показатель степени равен 0



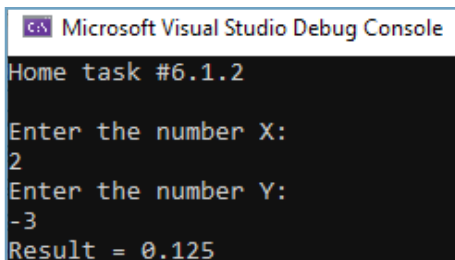
```
Microsoft Visual Studio Debug Console

Home task #6.1.2

Enter the number X:
5
Enter the number Y:
0
Result = 1
```

Рисунок 2

- Тест 2 — отрицательный показатель степени

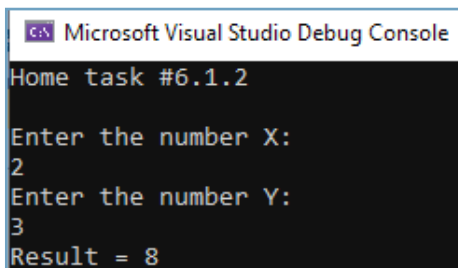


```
C:\> Microsoft Visual Studio Debug Console
Home task #6.1.2

Enter the number X:
2
Enter the number Y:
-3
Result = 0.125
```

Рисунок 3

- Тест 3 — положительный показатель степени

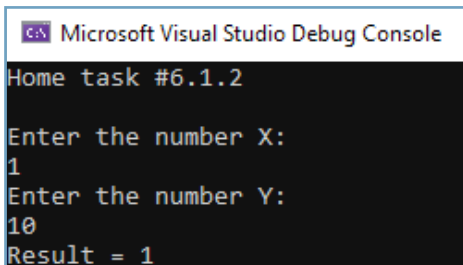


```
C:\> Microsoft Visual Studio Debug Console
Home task #6.1.2

Enter the number X:
2
Enter the number Y:
3
Result = 8
```

Рисунок 4

- Тест 4 — возведение числа 1 в степень

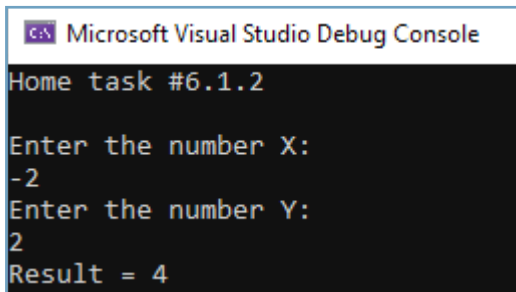


```
C:\> Microsoft Visual Studio Debug Console
Home task #6.1.2

Enter the number X:
1
Enter the number Y:
10
Result = 1
```

Рисунок 5

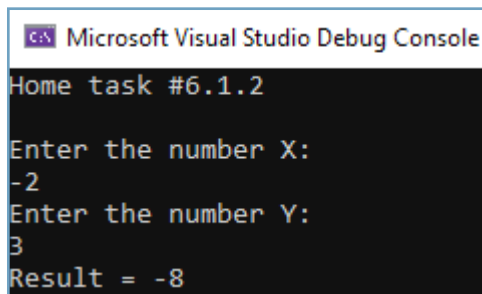
- Тест 5 — возведение отрицательного числа в четную степень.



```
Microsoft Visual Studio Debug Console
Home task #6.1.2
Enter the number X:
-2
Enter the number Y:
2
Result = 4
```

Рисунок 6

- Тест 6 — возведение отрицательного числа в нечетную степень.



```
Microsoft Visual Studio Debug Console
Home task #6.1.2
Enter the number X:
-2
Enter the number Y:
3
Result = -8
```

Рисунок 7

РЕШЕНИЕ ЗАДАНИЯ 3

Описание решения

Первый шаг при вычислении среднего арифметического чисел некоторого диапазона — это вычисление суммы чисел в заданном диапазоне

Пусть требуется вычислить сумму:

$$S = a_1 + a_2 + \dots + a_n.$$

Решение сводится к последовательному вычислению промежуточных сумм:

$$S_0 = 0,$$

(начальное значение, устанавливается перед вычислениями);

$$S_1 = S_0 + a_1;$$

$$S_2 = S_1 + a_2;$$

...

$$S_n = S_{n-1} + a_n = a_1 + a_2 + \dots + a_n;$$

Вычисление значения S_n представляет собой искомую сумму S . Значение промежуточных сумм S_1, \dots, S_{n-1} не требуется сохранять, поэтому последовательность вычислений, представленную выше, можно сформулировать в виде общей формулы:

$$S = S + a_i, \tag{1}$$

где a_i — слагаемое на i — том шаге.

Таким образом, вычисление суммы сводится к ее накоплению в переменной S на каждом шаге цикла.

Например, необходимо найти сумму чисел от 2 до 5 включительно.

Наши слагаемые:

$$a_1 = 2;$$

$$a_2 = 3;$$

$$a_3 = 4;$$

$$a_4 = 5;$$

$$S = a_1 + a_2 + a_3 + a_4;$$

$$S = 2 + 3 + 4 + 5.$$

Перед началом вычислений суммы еще нет, т.е. ее значение равно нулю.

$$S_0 = 0;$$

$$S_1 = S_0 + a_1 = 0 + 2 = 2;$$

$$S_2 = S_1 + a_2 = 2 + 3 = 5;$$

$$S_3 = S_2 + a_3 = 5 + 4 = 9;$$

$$S_4 = S_3 + a_4 = 9 + 5 = 14.$$

После получения суммы S выполняем ее деление на количество элементов диапазона, которое в данной задаче равно 1000.

$$\text{Среднее арифметическое} = S / 1000 \quad (2)$$

В нашем примере количество элементов (чисел для суммирования) равно 4, т. е.:

$$\text{Среднее арифметическое} = 14 / 4 = 3.5.$$

Решение

1. Выводим строку в консоль с описанием названия программы (по желанию, данная строка кода может быть пропущена), создаем одну переменную вещественного типа для хранения результата.

Устанавливаем начальное значение переменной результата в ноль (значение начальной суммы).

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

int main()
{
    cout << "Home task #6.1.3\n\n";
    float result;
    result = 0;

    return 0;
}
```

2. Организуем цикл: будем накапливать сумму (продолжать сложение), пока текущее число (которое приставляет счетчик цикла), двигаясь от начального значения (1) не достигнет значение 1000.

Будем использовать цикл `for`, так как известно необходимое количество итераций (диапазон изменения переменной — счетчика от 1 до 100).

В теле цикла находится только одна операция — накопление суммы по формуле (1).

После завершения цикла сумма готова, используем ее для вычисления среднего арифметического по формуле (2).

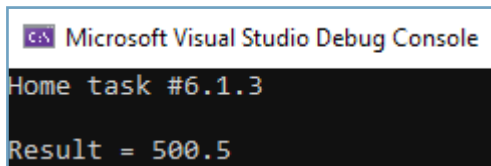
Выводим значение полученного результата пользователю в консоль.

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

int main()
```

```
{  
    cout << "Home task #6.1.3\n\n";  
    float result;  
  
    result = 0;  
  
    for (int i = 1; i <= 1000; i++)  
    {  
        result = result + i;  
    }  
    result = result / 1000;  
    cout << "Result = " << result;  
  
    return 0;  
}
```

Результаты работы программы (в консоли):



Microsoft Visual Studio Debug Console

```
Home task #6.1.3  
  
Result = 500.5
```

Рисунок 8