



Муниципальное бюджетное образовательное учреждение дополнительного образования  
**«Гуманитарный центр интеллектуального развития»**  
городского округа Тольятти

---



## **СБОРНИК СПРАВОЧНЫХ МАТЕРИАЛОВ «ПАМЯТКИ ПРОГРАММИСТА С++»**

Учебное пособие для обучающихся по дополнительной программе  
«Программирование - это интересно: С++»

**Разработчик:**  
Савина Дарья Александровна,  
педагог дополнительного образования  
МБОУ ДО ГЦИР

Тольятти, 2024

## ОГЛАВЛЕНИЕ

ВВЕДЕНИЕ .....	2
1. ПАМЯТКА ПО ВВОДУ-ВЫВОДУ в C++ .....	3
2. ПАМЯТКА «АЛГОРИТМ НАПИСАНИЯ КОМПЬЮТЕРНОЙ ПРОГРАММЫ» .....	4
3. ПАМЯТКА ПО УСЛОВНЫМ КОНСТРУКЦИЯМ.....	7
4. УСЛОВНАЯ КОНСТРУКЦИЯ SWITCH – CASE .....	8
5. ПАМЯТКА ДЛЯ УЧАЩИХСЯ «ЦИКЛЫ» .....	9
6. ПОЛЬЗОВАТЕЛЬСКИЕ ФУНКЦИИ В ПРОГРАММИРОВАНИИ .....	10
7. ТАБЛИЦА ASCII .....	11
СПИСОК ИСПОЛЬЗУЕМОЙ ЛИТЕРАТУРЫ .....	12

## ВВЕДЕНИЕ

Сборник справочных материалов для учащихся «Памятки программиста C++» предназначен для учащихся 11-13 лет, обучающихся по программе «Программирование – это интересно. C++».

Памятки содержат информацию, которая помогает учащимся актуализировать базовые знания, а также во время решения задач в большей степени концентрироваться на логике решения задачи, и в меньшей степени на знании синтаксиса. Памятка содержит схемы базовых алгоритмических конструкций программирования (линейный, разветвляющийся, циклический алгоритм), а также схему объявления пользовательских функций и таблицу кодировки ASCII.

Содержание памятки:

- Памятка по вводу-выводу
- Алгоритм написания программы (линейный алгоритм)
- Памятка по условным конструкциям
- Памятка по условной конструкции switch - case
- Памятка по циклам
- Памятка по пользовательским функциям
- Таблица ASCII

Памятка используется учащимися как ознакомительного, так и базового уровня обучения, т.к. содержит справочную информацию по основным конструкциям, используемым при решении практически любых задач по любым темам.

## 1. ПАМЯТКА ПО ВВОДУ-ВЫВОДУ В C++

Типы переменных	<b>int</b> – целое число <b>string</b> - строка
Обозначение переменных, присвоение значения	<b>int number;</b> <b>int number = 5;</b>  <b>std::string name;</b> <b>std::string name = "Мария";</b>
Вывод данных (любое сообщение, которое пользователь должен увидеть на экране)	<b>std::cout &lt;&lt; "Как тебя зовут?\n";</b>  <b>\n</b> добавляется, если нужно перейти на следующую строку. Добавлять не обязательно. Но если добавляется, то только к тому тексту, что пишется в " ". Также для перехода на следующую строку можно добавить endl с оператором вывода:  <b>std::cout &lt;&lt; "Как тебя зовут?" &lt;&lt; std::endl;</b>  Вывод значения переменных: <b>std::cout &lt;&lt; name;</b>  Вывод текста + значение переменных: <b>std::cout &lt;&lt; "Тебя зовут " &lt;&lt; name;</b>
Ввод данных (то, что пользователь вводит с клавиатуры, и это потом «записывается» в переменную)	<b>std::cin &gt;&gt; name;</b>

\* **std::** не добавляется перед cout, cin, string и endl, если над функцией main мы указываем, что используем стандартное пространство имен:  
using namespace std

## 2. ПАМЯТКА «АЛГОРИТМ НАПИСАНИЯ КОМПЬЮТЕРНОЙ ПРОГРАММЫ»

Шаг	Что делать
<b>Шаг 1.</b> <b>Подготовка</b> <b>рабочей среды</b>	<p>Откройте Visual Studio.</p> <p>Удалите все закомментированные строки (строки, выделенные зеленым цветом и начинающиеся с //). Это комментарии – они никак не обрабатываются компилятором и не участвуют в коде).</p> <p>Ваша рабочая среда должна выглядеть так:</p> <pre>#include &lt;iostream&gt;  int main() {  //Здесь может быть пусто или может по умолчанию стоять строка "Hello world". Ее можно удалить или оставить для дальнейшей работы, если это нужно. }</pre>
<b>Шаг 2.</b> <b>Установка</b> <b>русской</b> <b>кодировки</b>	<p>В файле «Русская кодировка» на рабочем столе в папке C++ нужно скопировать строку и вставить ВНУТРЬ функции main.</p> <pre>int main() {  setlocale(LC_ALL, "Russian"); //Внутри функции Main – это значит СЮДА – между двух фигурных скобок первой строкой. //////////ВСЯ ПРОГРАММА ПИШЕТСЯ ЗДЕСЬ – ВНУТРИ ФУНКЦИИ Main////////// }</pre>
<b>Шаг 3.</b> <b>Подключения</b> <b>стандартного</b> <b>пространства</b> <b>имён</b>	<p>Чтобы не писать перед каждой командой ввода, вывода и многих других ключевых слов C++, входящий в стандартное пространство имён, <b>std::</b> можно подключить заголовочный файл «использование стандартного пространства имён». Делается это после подключения потока ввода и вывода: <code>include &lt;iostream&gt;</code>. Не забываем про точку с запятой в конце!!!</p> <pre>#include &lt;iostream&gt; using namespace std;  int main() {</pre>

	}
<p><b>Шаг 4.</b>  <b>Начинаем</b>  <b>писать</b>  <b>программу.</b>  <b>Объявление</b>  <b>переменных</b></p>	<p>ВНИМАТЕЛЬНО прочитайте задачу и определите, что от вас хотят. ВСЕ данные, с которыми вы будете работать, должны лежать в ПЕРЕМЕННЫХ.</p> <p><b>На этом этапе вы должны понять:</b></p> <ul style="list-style-type: none"> <li>- какой тип ваших переменных (число – <b>int</b>, строка (набор символов) - <b>string</b>). Как понять, какой тип? Очень просто! Еще раз ВНИМАТЕЛЬНО посмотрите задачу. Если с вас спрашивают посчитать что-то: возраст, рост, длину, ширину, расстояние, скорость, просто числа и вообще что угодно что выражается ЧИСЛАМИ – это переменные типа <b>int</b></li> <li>Если с вас спрашивают имя, фамилию, название страны, ваше настроение, любимый цвет, то есть что угодно что вы бы написали БУКВАМИ – это тип строка <b>string</b>.</li> <li>- сколько этих переменных. Как это понять? Посчитать. Если с вас спрашивается: посчитайте сумму двух чисел, переменных как минимум должно быть две, потому что числе два. Если у вас задача для вычисления скорости, времени или расстояния, у вас должно быть три переменных для скорости, времени и расстояния соответственно. <b>ВСЕ</b> числа или строки, с которыми вы работаете, должны лежать в отдельных переменных.</li> </ul> <p><b>ДАЛЕЕ ВЫ ОБЪЯВЛЯЕТЕ ПЕРЕМЕННЫЕ.</b></p> <p>Как это сделать? Указать сначала тип, потом имя. Имя вы придумываете сами, записывая его латинскими буквами без пробелов. Если вы изначально знаете, чему равна переменная, можно указать ее значение через оператор присваивания (=). Если вы не знаете это значение (его, например, должен ввести пользователь), то просто указываете тип, имя и точку с запятой.</p> <p><b><u>Примеры правильного объявления типа int «целое число»</u></b></p> <pre>int a; int c = 5; int b = a + c;</pre> <p><b><u>Примеры правильного объявления типа string «строка»</u></b></p> <pre>string name; string name = "Maria";</pre>
<p><b>Шаг 5.</b>  <b>Пишем саму</b>  <b>программу.</b></p>	<p>Чаще всего программа начинается с инструкции пользователю: что ему делать? Ввести числа, свой рост, свой возраст, свое имя (читаете в условии задачи). Чтобы написать инструкцию, используем операторы ВЫВОДА (вывод – это когда на черном экране (в консоли) появляется некий текст, который вы видите):</p>

	<p><code>cout &lt;&lt; “Здесь вы пишете то, что хотите, чтобы пользователь сделал. Например, ввел числа”;</code></p> <p>После этого чаще всего пользователь должен ввести то, что вы его попросили. Для этого используем операторы ВВОДА (Ввод – это когда вы своими руками пишете с клавиатуры ответы на то, что вас спрашивает программа: число, имя, возраст и т.п.).</p> <p><code>cin &gt;&gt; name;</code></p> <p>Вместо переменной <code>name</code> здесь должна стоять та переменная, которую вы изначально объявили (см пункт 4) и в которую вы положите конкретный ответ.</p> <p><b><u>ВЫВОД ПЕРЕМЕННОЙ:</u></b></p> <p><code>cout &lt;&lt; name;</code></p> <p><b><u>ВЫВОД ПЕРЕМЕННОЙ + ПОЯСНЯЮЩЕГО ТЕКСТА:</u></b></p> <p><code>cout &lt;&lt; “Сумма чисел равна ” &lt;&lt; a;</code></p>
<p><b>Шаг 6.</b> <b>Работаем с переменными</b></p>	<p>То, что делать дальше зависит от задачи. Нужно произвести какие-то операции с переменными. Например, математические операции.</p> <p>С переменными типа <code>int</code> могут быть произведены любые арифметические операции (сложение, вычитание, умножение, деление и т.п.).</p> <p style="text-align: center;"><b><u>ПРИМЕР:</u></b></p> <pre> #include &lt;iostream&gt; using namespace std;  int main() { setlocale(LC_ALL, "Russian");  int a, b, c; cout &lt;&lt; “Введите число: “; cin &gt;&gt; a ; cout &lt;&lt; “Введите число: “; cin &gt;&gt; b ; c = a + b; cout &lt;&lt; “Сумма чисел равна “ &lt;&lt; c; } </pre>

### 3. ПАМЯТКА ПО УСЛОВНЫМ КОНСТРУКЦИЯМ

Сокращенная форма условной конструкции	<pre>if (условие) { Блок действий, которые выполняются, если условие истинно; }</pre>
Полная форма условной конструкции	<pre>if (условие) { Блок действий, которые выполняются, если условие истинно; } else { Блок действий, которые выполняются при альтернативном варианте (все остальные случаи, кроме тех, что описаны в условии if); }</pre>
Несколько условий в одной конструкции	<pre>if (условие1) { Блок действий, которые выполняются, если условие 1 истинно; } else if (условие2) { Блок действий, которые выполняются, если условие 2 истинно; } else { Блок действий, которые выполняются при альтернативном варианте (все остальные случаи, кроме тех, что описаны в условиях if); }</pre>

## 4. УСЛОВНАЯ КОНСТРУКЦИЯ SWITCH – CASE

```
switch(выражение)
{
    case значение_1: инструкции_1;
    case значение_2: инструкции_2;
    .....
    case значение_N: инструкции_N;

    default: инструкции;
}
```

Пример. Калькулятор с применением конструкции switch case:

```
float x, y, z;
char op;
cin >> x >> y;
cin >> op;
    switch (op) {
        case '+': {
            z = x + y;
            break;
        }
        case '-': {
            z = x - y;
            break;
        }
        case '*': {
            z = x * y;
            break;
        }
        case '/': {
            z = x / y;
            break;
        }
        default: {
            cout << "Error" << endl;
        }
    }
    cout << z << endl;
```

## 5. ПАМЯТКА ДЛЯ УЧАЩИХСЯ «ЦИКЛЫ»

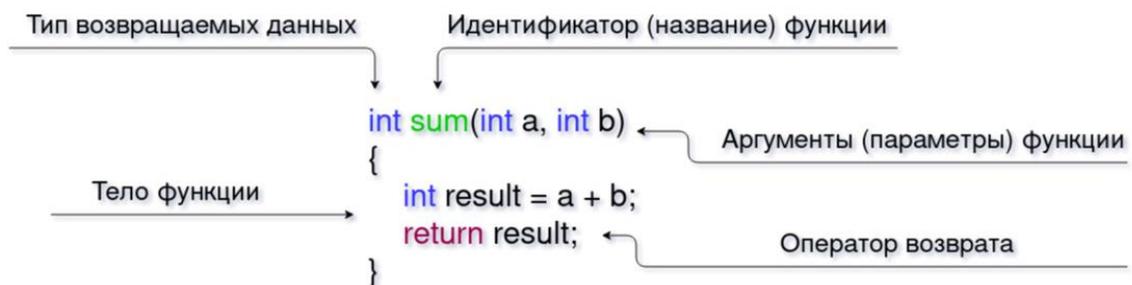
<i>Название цикла</i>	<i>Содержание</i>
Цикл <b>while</b>	<p><b>while (условие)</b>  {  Блок действий, который <u>повторяется</u>, пока условие истинно;  }    <u>Пример (вывод на экран фразы “Hello” 5 раз, если переменная x изначально равна 0 ):</u></p> <pre>while (x &lt; 5) { cout &lt;&lt; “Hello” &lt;&lt;endl; x+=1; }</pre>
Цикл <b>do while</b>	<p><b>do</b>  {  Блок действий, который <u>повторяется</u>, пока условие истинно;  }  <b>while ( условие);</b>    <u>Пример (вывод на экран фразы “Hello” 5 раз, если переменная x изначально равна 0 ):</u></p> <pre>do { cout &lt;&lt; “Hello” ” &lt;&lt;endl; x+=1; } while (x &lt; 5);</pre>
Цикл <b>for</b>	<p><b>for (инициализация счётной переменной; условие; приращение переменной)</b>  {  Блок действий, который <u>повторяется</u> столько раз, сколько указано в условии;  }    <u>Пример (вывод на экран фразы “Hello” 5 раз:</u></p> <pre><u>for(int I = 0; i&lt;5; i++)</u> { cout &lt;&lt; “Hello” ” &lt;&lt;endl; }</pre>

## 6. ПОЛЬЗОВАТЕЛЬСКИЕ ФУНКЦИИ В ПРОГРАММИРОВАНИИ

### Определение функции

```
типВозвращаемогоЗначения имяФункции (параметры если необходимы)
{
    код (тело функции); // один или несколько операторов
    return значение;
}
```

### Пример функции сложения двух чисел



### Пример определения и вызова функции

```
1  #include <iostream>
2
3  int add(int x, int y)
4  {
5      ...
6      return x + y;
7  }
8  возвращаемое значение = 9
9
10 int main()
11 {
12     ...
13     std::cout << add(4, 5) << std::endl;
14     return 0;
15 }
```

The diagram shows the code from the previous block with annotations:

- x = 4:** Points to the value `4` in the function call `add(4, 5)`.
- y = 5:** Points to the value `5` in the function call `add(4, 5)`.
- возвращаемое значение = 9:** Points to the `return` statement in the `add` function definition.

## 7. ТАБЛИЦА ASCII

**ASCII** (American Standard Code for Information Interchange — Стандартный американский код обмена информацией) — это код для представления символов в виде чисел, в котором каждому символу сопоставлено число от 0 до 127.

В большинстве компьютеров код ASCII используется для представления текста, что позволяет передавать данные от одного компьютера на другой.

Коды от 0 до 32 – непечатаемые, служебные символы

Коды от 128 до 255 представляют собой расширение таблицы ASCII. Эти коды используются для кодирования символов национальных алфавитов, а также символов псевдографики.

32 пробел	48 0	64 @	80 P	96 `	112 p
33 !	49 1	65 A	81 Q	97 a	113 q
34 "	50 2	66 B	82 R	98 b	114 r
35 #	51 3	67 C	83 S	99 c	115 s
36 \$	52 4	68 D	84 T	100 d	116 t
37 %	53 5	69 E	85 U	101 e	117 u
38 &	54 6	70 F	86 V	102 f	118 v
39 ' .	55 7	71 G	87 W	103 g	119 w
40 (	56 8	72 H	88 X	104 h	120 x
41 )	57 9	73 I	89 Y	105 i	121 y
42 * .	58 :	74 J	90 Z	106 j	122 z
43 +	59 ;	75 K	91 [	107 k	123 {
44 ,	60 <	76 L	92 \	108 l	124
45 -	61 =	77 M	93 ]	109 m	125 }
46 .	62 >	78 N	94 ^	110 n	126 ~
47 /	63 ?	79 O	95 _	111 o	127

## СПИСОК ИСПОЛЬЗУЕМОЙ ЛИТЕРАТУРЫ

1. Орленко, П.А. С++ на примерах. Практика, практика и только практика. / П.А.Орленко, П.В.Евдокимов. – СПб. : Наука и техника, 2022. – 288 с.
2. Доусон, М. Изучаем С++ через программирование игр. / Майкл Доусон; пер. с англ. Е.Зазноба, О.Сивченко – СПб.: Питер, 2022. – 352 с.