



## §. 基础知识题 – 字符的输入与输出

要求:

- 1、完成本文档中所有的题目并写出分析、运行结果
- 2、无特殊说明，均使用VS2022编译即可
- 3、直接在本文件上作答，**写出答案/截图（不允许手写、手写拍照截图）**即可；填写答案时，为适应所填内容或贴图，**允许调整**页面的字体大小、颜色、文本框的位置等
  - ★ 贴图要有效部分即可，不需要全部内容
  - ★ 在保证一页一题的前提下，具体页面布局可以自行发挥，简单易读即可
  - ★ **不允许**手写在纸上，再拍照贴图
  - ★ **允许**在各种软件工具上完成（不含手写），再截图贴图
  - ★ 如果某题要求VS+Dev的，则如果两个编译器运行结果一致，贴VS的一张图即可，如果不一致，则两个图都要贴
- 4、转换为pdf后提交
- 5、**3月21日前**网上提交本次作业（在“文档作业”中提交）



## §. 基础知识题 - 字符的输入与输出

贴图要求：只需要截取输出窗口中的有效部分即可，如果全部截取/截取过大，则视为无效贴图

例：无效贴图

```
Microsoft Visual Studio 调试控制台  
Hello, world!  
D:\Workspace\VS2019-Demo\Debug\cpp-demo.exe (进程 7484)已退出, 代码为 0。  
按任意键关闭此窗口. . .
```

例：有效贴图

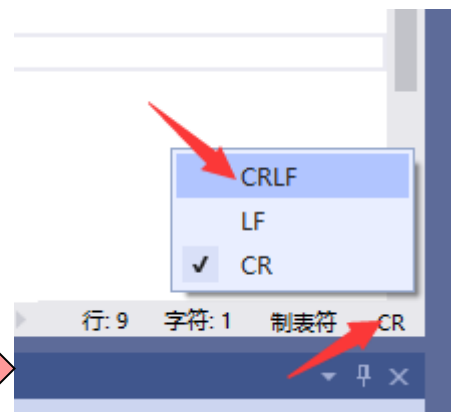
```
Microsoft Visual Studio 调试控制台  
Hello, world!
```



## §. 基础知识题 - 字符的输入与输出

附：用WPS等其他第三方软件打开PPT，将代码复制到VS2022中后，如果出现类似下面的**编译报错**，则观察源程序编辑窗

的右下角是否为CR，如果是，单击CR，在弹出中选择CRLF，再次CTRL+F5运行即可



## §. 基础知识题 - 字符的输入与输出



特别提示:

- 1、做题过程中，先按要求输入，如果想替换数据，也要先做完指定输入
- 2、如果替换数据后出现某些问题，先记录下来，不要问，等全部完成后，还想不通再问(也许你的问题在后面的题目中有答案)
- 3、不要偷懒、不要自以为是的脑补结论!!!
- 4、先得到题目要求的小结论，再综合考虑上下题目间关系，得到综合结论
- 5、这些结论，是让你记住的，不是让你完成作业后就忘掉了
- 6、换位思考(从老师角度出发)，这些题的目的是希望掌握什么学习方法？



## §. 基础知识题 – 字符的输入与输出

基本知识点:

- 1、cin和getchar的区别: cin是按格式读入, 到空格、回车、非法为止; getchar是只读一个字符
- 2、两者的共同点: 都有输入缓冲区, 输入必须以回车结束, 从输入缓冲区去取得需要的内容后, 多余的内容还放在输入缓冲区中, 等到下次读入 (如果程序结束, 则操作系统会清空输入缓冲区)
- 3、\_getche()/\_getch() 是没有输入缓冲区的, 输入后不需要按回车键
- 4、getchar() 的返回是int型, 因为除了正常的256个ASCII字符 (含基本和扩展ASCII码、中文、其它语言文字等), 还需要额外考虑一个输入出错情况下的返回, 因此无法用1字节返回值

# 5、先认真看课件!!!



## §. 基础知识题 - 字符的输入与输出

### 1、 putchar的基本使用

字符输出函数putchar的基本知识:

形式: putchar(字符变量/常量)

功能: 输出一个字符

```
char a='A';
```

```
putchar(a);
```

```
putchar('A');
```

```
putchar('\x41');
```

```
putchar('\101');
```

均表示输出'A'

★ 某些编译器需要 #include <cstdio> 或 #include <stdio.h> (目前所用的双编译器均不需要)

★ 返回值是int型, 是输出字符的ASCII码, 可赋值给字符型/整型变量

本页不用作答



## §. 基础知识题 - 字符的输入与输出

### 1、putchar的基本使用

A. 程序如下，观察编译及运行结果（贴图在清晰可辨的情况下尽可能小）

```
#include <iostream>
#include <stdio>
using namespace std;
int main()
{
    char ret1;
    cout << (ret1 = putchar('A')) << endl;
    int ret2;
    cout << (ret2 = putchar('B')) << endl;
    return 0;
}
```

1、观察运行结果



2、分析运行结果中各输出是哪个语句/函数造成的  
(可选: cout/putchar)

putchar

3、这个例子能确认上个Page的基本知识中的说法:  
“返回值是int型, 是输出字符的ASCII码”  
完全正确/部分正确吗?

部分正确



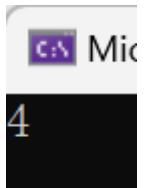
## §. 基础知识题 – 字符的输入与输出

### 1、putchar的基本使用

B. 自行构造测试程序，证明putchar的返回值是int型而不是char型(要求两种方法，可以从课件找，也可以自行构造)

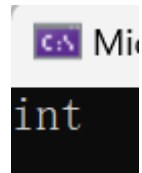
//方法一

```
✓#include <iostream>
| #include <cstdio>
| using namespace std;
✓int main()
| {
|     cout << sizeof(putchar('A')) << endl;
| }
|
```



//方法2

```
✓#include <iostream>
| #include <cstdio>
| using namespace std;
✓int main()
| {
|     cout << typeid(putchar('A')).name() << endl;
| }
|
```







## §. 基础知识题 - 字符的输入与输出

此页不要删除，也没有意义，仅仅为了分隔题目



## §. 基础知识题 - 字符的输入与输出

### 2、getchar的基本使用

字符输入函数getchar的基本知识:

形式: `getchar()`

功能: 输入一个字符 (给指定的变量)

- ★ 某些编译器需要 `#include <cstdio>` 或 `#include <stdio.h>` (目前所用的双编译器均不需要)
- ★ 返回值是int型, 是输入字符的ASCII码, 可赋值给字符型/整型变量
- ★ 输入有回显, 而且不是键盘输入一个字符后立即执行`getchar`, 必须要等按回车后才执行  
(弄清楚上课课件中的输入缓冲区的概念)
- ★ 可以输入空格, 回车等`cin`无法处理的非图形字符, 但仍不能处理转义符
- ★ `getchar/cin`等每次仅从输入缓冲区中取需要的字节, 多余的字节仍保留在输入缓冲区中供下次读取

本页不用作答



# §. 基础知识题 - 字符的输入与输出

## 2、getchar的基本使用

A. 程序如下，观察编译及运行结果（可手填，如果贴图，要求在清晰可辨的情况下尽可能小）

<pre>#include &lt;iostream&gt; #include &lt;cstdio&gt; using namespace std; int main() {     char ch;     ch = getchar();     cout &lt;&lt; ch &lt;&lt; endl;      return 0; }</pre>	<pre>#include &lt;iostream&gt; #include &lt;cstdio&gt; using namespace std; int main() {     char ch;     cout &lt;&lt; (ch = getchar()) &lt;&lt; endl;      return 0; }</pre>	<pre>#include &lt;iostream&gt; #include &lt;cstdio&gt; using namespace std; int main() {     int ch;     ch = getchar();     cout &lt;&lt; ch &lt;&lt; endl;      return 0; }</pre>
输入：a✓ 输出： ____a____  输出的是：赋值表达式值 (ch的值/赋值表达式值)	输入：a✓ 输出： ____a____  输出的是：赋值表达式值 (ch的值/赋值表达式值)	输入：a✓ 输出： ____97____



## §. 基础知识题 – 字符的输入与输出

### 2、getchar的基本使用

B. 自行构造测试程序，证明getchar的返回值是int型而不是char型(要求两种方法，可以从课件找，也可以自行构造)

//方法一

```
✓#include <iostream>
[ #include <cstdio>
  using namespace std;
✓int main()
{
  cout << typeid(getchar()).name() << endl;
  return 0;
}
```

Microsoft Visual Studio 调试控制台

int

//方法2

```
✓#include <iostream>
[ #include <cstdio>
  using namespace std;
✓int main()
{
  cout << sizeof(getchar()) << endl;
  return 0;
}
```

Microsoft Visual Studio 调试控制台

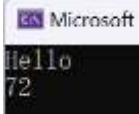
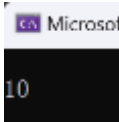
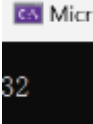
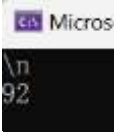
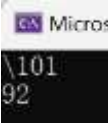
4



# §. 基础知识题 - 字符的输入与输出

## 2、getchar的基本使用

C. 程序如下，观察编译及运行结果（可手填，如果贴图，要求在清晰可辨的情况下尽可能小）

<pre>#include &lt;iostream&gt; #include &lt;cstdio&gt; using namespace std; int main() {     char ch;     ch = getchar();     cout &lt;&lt; int(ch) &lt;&lt; endl;      return 0; }</pre>	<div>1、键盘输入：Hello✓（5个字母+回车）</div> <div>2、键盘输入：✓（空回车）</div> <div>3、键盘输入：␣✓（空格+回车）</div> <div>4、键盘输入：\n✓（2个字符+回车）</div> <div>5、键盘输入：\101✓（4个字符+回车）</div> <div>结论：可以输入__a)__、__c)__等cin无法处理的非图形字符，但仍不能处理__b)__ a) 空格 b) 转义符 c) 回车</div>
---	---



## §. 基础知识题 – 字符的输入与输出

### 2、getchar的基本使用

D. 程序如下，观察编译及运行结果（可手填，如果贴图，要求在清晰可辨的情况下尽可能小）

```
#include <iostream>
#include <cstdio>
using namespace std;
int main()
{
    cout << "--Step1--" << endl;
    cout << getchar() << endl;

    cout << "--Step2--" << endl;
    cout << getchar() << endl;

    cout << "--Step3--" << endl;
    cout << getchar() << endl;

    cout << "--Step4--" << endl;
    cout << getchar() << endl;
    return 0;
}
```

本次要求仔细观察运行现象及结果，特别是Step1~4出现的时机!!!

1、每次输入一个回车

程序从开始执行到结束，共停顿了\_\_4\_\_次来等待输入

第1次停顿时，屏幕上输出的最后一行是Step\_\_1\_\_ ?

第2次停顿时，屏幕上输出的最后一行是Step\_\_2\_\_ ? (没有则不填)

第3次停顿时，屏幕上输出的最后一行是Step\_\_3\_\_ ? (没有则不填)

第4次停顿时，屏幕上输出的最后一行是Step\_\_4\_\_ ? (没有则不填)

2、第一次输入一个字母+回车，以后每次停顿，均输入一个字母+回车

程序从开始执行到结束，共停顿了\_\_2\_\_次来等待输入

第1次停顿时，屏幕上输出的最后一行是Step\_\_1\_\_ ?

第2次停顿时，屏幕上输出的最后一行是Step\_\_3\_\_ ? (没有则不填)

第3次停顿时，屏幕上输出的最后一行是Step\_\_\_\_\_ ? (没有则不填)

第4次停顿时，屏幕上输出的最后一行是Step\_\_\_\_\_ ? (没有则不填)

3、第一次即输入4个以上的字母+回车

程序从开始执行到结束，共停顿了\_\_1\_\_次来等待输入

第1次停顿时，屏幕上输出的最后一行是Step\_\_1\_\_ ?

第2次停顿时，屏幕上输出的最后一行是Step\_\_\_\_\_ ? (没有则不填)

第3次停顿时，屏幕上输出的最后一行是Step\_\_\_\_\_ ? (没有则不填)

第4次停顿时，屏幕上输出的最后一行是Step\_\_\_\_\_ ? (没有则不填)

结论：getchar每次仅从输入缓冲区中取需要的字节，多余的字节仍保留在输入缓冲区中供下次读取

思考：结合“cin与cout的基本使用”中3.c的例子，考虑一下3.c中非法m对int的影响(错在第几个数)与输入缓冲区的关系，为什么？cin将数据储存在输入缓冲区，输出每次读取四个字节，非法字符无法被读取，则输出时输出错误数据。



## §. 基础知识题 – 字符的输入与输出

### 2、getchar的基本使用

#### E. 自行构造证明D结论的使用cin读入的测试程序

```
#include <iostream>
using namespace std;
int main()
{
    return 0;
}
```

```
✓ #include <iostream>
  | #include <cstdio>
  | using namespace std;
✓ int main()
  | {
  |     char a;
  |     cout << "--Step1--" << endl;
  |     cin >> a;
  |     cout << a << endl;
  |
  |     cout << "--Step2--" << endl;
  |     cin >> a;
  |     cout << a << endl;
  |
  |     cout << "--Step3--" << endl;
  |     cin >> a;
  |     cout << a << endl;
  |
  |     cout << "--Step4--" << endl;
  |     cin >> a;
  |     cout << a << endl;
  |     return 0;
  | }
```

本次要求仔细观察运行现象及结果，特别是Stepx出现的时机!!!

因为cin不能读取空格、回车（有特殊方法可读，先忽略），因此测试有所不同

- 1、第一次输入两个字母+回车，以后每次停顿，均输入两个字母+回车  
程序从开始执行到结束，共停顿**2**次来等待输入  
第1次停顿时，屏幕上输出的最后一行是Step**1**?  
第2次停顿时，屏幕上输出的最后一行是Step**4**?（没有则不填）  
第3次停顿时，屏幕上输出的最后一行是Step\_\_\_\_\_ ?（没有则不填）  
第4次停顿时，屏幕上输出的最后一行是Step\_\_\_\_\_ ?（没有则不填）
- 2、第一次即输入4个以上的字母+回车  
程序从开始执行到结束，共停顿**1**次来等待输入  
第1次停顿时，屏幕上输出的最后一行是Step**1**?  
第2次停顿时，屏幕上输出的最后一行是Step\_\_\_\_\_ ?（没有则不填）  
第3次停顿时，屏幕上输出的最后一行是Step\_\_\_\_\_ ?（没有则不填）  
第4次停顿时，屏幕上输出的最后一行是Step\_\_\_\_\_ ?（没有则不填）

结论：cin每次仅从输入缓冲区中取需要的字节，多余的字节仍保留在输入缓冲区中供下次读取



## §. 基础知识题 - 字符的输入与输出

此页不要删除，也没有意义，仅仅为了分隔题目





# §. 基础知识题 - 字符的输入与输出

## 3、getchar、\_getch与\_getche的基本使用

- 1、测试时cmd窗口下面不能是中文输入法
- 2、<conio.h>是\_getch()/\_getche()需要的头文件

A. 程序如下，观察编译及运行结果（贴图在清晰可辨的情况下尽可能小）

<pre>#include &lt;iostream&gt; using namespace std;  int main() {     char ch;     ch = getchar();     cout &lt;&lt; (int)ch &lt;&lt; endl;      return 0; }</pre>	<pre>#include &lt;iostream&gt; #include &lt;conio.h&gt; using namespace std;  int main() {     char ch;     ch = _getch();     cout &lt;&lt; (int)ch &lt;&lt; endl;      return 0; }</pre>	<pre>#include &lt;iostream&gt; #include &lt;conio.h&gt; using namespace std;  int main() {     char ch;     ch = _getche();     cout &lt;&lt; (int)ch &lt;&lt; endl;      return 0; }</pre>
<div>1、输入：a</div> <div>输出：_____</div> <div>输入回显：___有___(有/无)</div> <div>按回车生效：___是___(是/否)</div> <div>2、输入：↵(直接回车)</div> <div>输出：_____</div>	<div>1、输入：a</div> <div>输出：_____</div> <div>输入回显：___无___(有/无)</div> <div>按回车生效：___否___(是/否)</div> <div>2、输入：↵(直接回车)</div> <div>输出：_____</div>	<div>1、输入：a</div> <div>输出：_____</div> <div>输入回显：___有___(有/无)</div> <div>按回车生效：___否___(是/否)</div> <div>2、输入：↵(直接回车)</div> <div>输出：_____</div>

本题要求  
VS+Dev

注：直接按回车时的差异，了解即可，具体原因有兴趣自己课外查阅，不提供技术支持

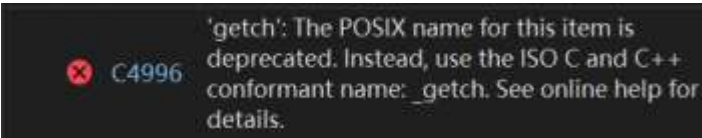
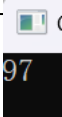
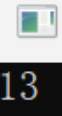

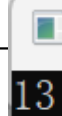


# §. 基础知识题 - 字符的输入与输出

3、getchar、\_getch与\_getche的基本使用

- 1、测试时cmd窗口下面不能是中文输入法
- 2、<conio.h>是\_getch()/\_getche()需要的头文件

B. 程序如下，观察编译及运行结果（贴图在清晰可辨的情况下尽可能小）

<div>本题要求VS+Dev</div> <div>哪个编译器报错？ VS报错 哪个编译器下结果同A？ Dev结果同A</div> <div></div>	<pre>#include &lt;iostream&gt; #include &lt;conio.h&gt; using namespace std;  int main() {     char ch;     ch = getch();     cout &lt;&lt; (int)ch &lt;&lt; endl;      return 0; }</pre>	<pre>#include &lt;iostream&gt; #include &lt;conio.h&gt; using namespace std;  int main() {     char ch;     ch = getche();     cout &lt;&lt; (int)ch &lt;&lt; endl;      return 0; }</pre>
	<div>1、输入：a ✓ 输出：  输入回显： ____无____ (有/无) 按回车生效： ____否____ (是/否)</div> <div>2、输入： ✓ (直接回车) 输出： </div>	<div>1、输入：a ✓ 输出：  输入回显： ____有____ (有/无) 按回车生效： ____否____ (是/否)</div> <div>2、输入： ✓ (直接回车) 输出： </div>



## §. 基础知识题 - 字符的输入与输出

此页不要删除，也没有意义，仅仅为了分隔题目