一、关于高级计算机语言和低级计算机语言的优劣，我认为主要体现在以下三点：

\1. 可读性：，高级语言因为有丰富的语法结构和标准库支持，如数据结构、函数、等，不需要关心底层的硬件细节，代码通常更容易阅读。而低级语言代码非常接近计算机的硬件和机器指令，往往晦涩难懂，可读性较差

\2. 执行速度及相应开销：高级语言编写的程序通常需要通过编译器或解释器转换成机器码才能执行，这可能会引入一定的性能开销，尽管现代编译器优化技术已经能够生成非常高效的代码，但在某些对性能要求极高的场景下可能仍相对缓慢。而低级语言编写的程序可以直接被计算机硬件执行，几乎没有额外的解释开销，因此在执行速度上有明显优势。

\3. 学习难度：对于初学者来说，高级语言更容易上手，入门门槛较低。丰富的文档和社区支持也有助于学习和解决问题。学习低级语言需要对计算机体系结构和硬件有深入的了解，难度较大。但掌握低级语言有助于深入理解计算机的工作原理。

二、解读代码内容

\1. #include <stdio.h>：这行代码是告诉编译器包含stdio.h的 头文件。这个头文件包含了进行输入输出操作的函数的声明。

\2. int main()：这是主函数的开始，C语言程序从这里开始执行。int表示这个函数返回一个整数值。

\3. printf("Hello, world!");：这行代码调用了printf()函数，它是标准输入输出库中的一个函数，用于打印””内的内容。在这个例子中，它打印出“Hello, world!”这句话。

\4. return 0;：这行代码表示main函数的结束，并返回一个整数值0给操作系统。

三、删去return 0;不会影响运行结果

四、int类型是计算机存储整型的方式，用int对main函数进行定义或说明意味着函数结尾需要返回一个整型值

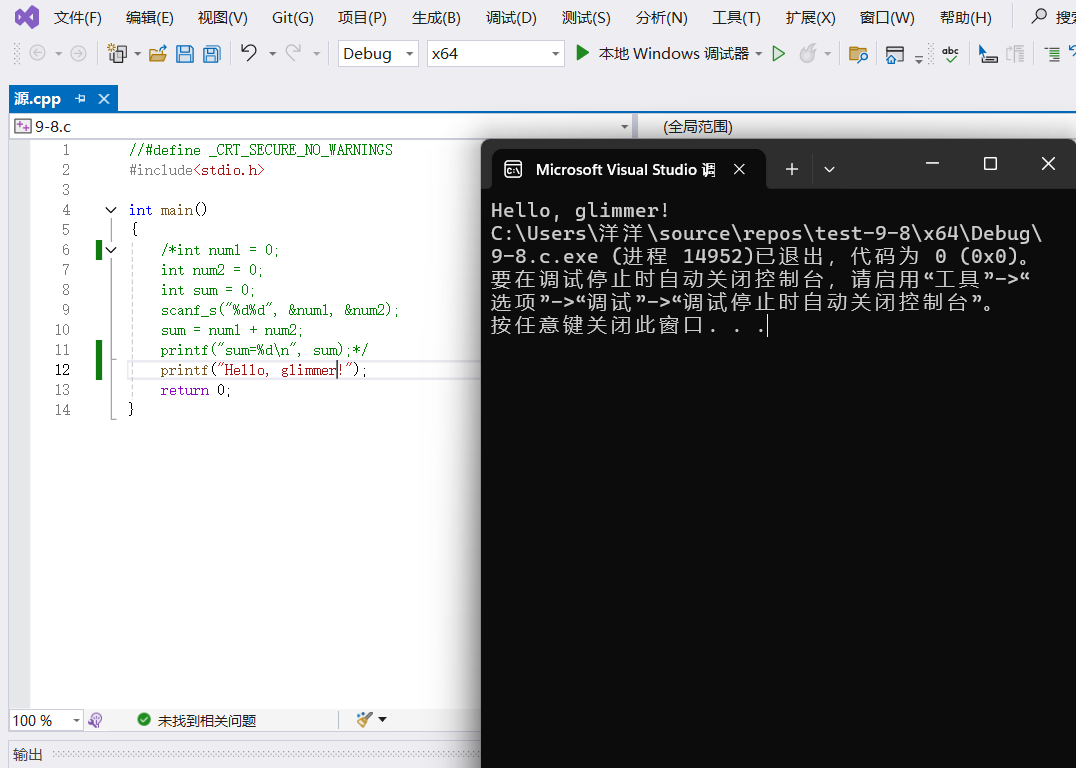
五、代码块：

~~~c#

\#include <stdio.h> int main() { printf("Hello, world!"); return 0; }

~~~

![image-20241019203512280](C:\Users\洋洋\AppData\Roaming\Typora\typora-user-images\image-20241019203512280.png)



~~~

六、对小明代码的修改

代码块：

~~~c

#include <stdio.h>

int main() {

int code;

printf("Show me your code,please.")；

while(1){

scanf("%d", &code);

if(code >= 100000 || code <= 999999)printf("I am super hacker!\n");

return 0;

else printf("Fake code!\n");

}

return 0;

}

~~~