第一次选择题

1

A: 变量定义：用于为变量分配存储空间，还可为变量指定初始值。

变量声明：用于向程序表明变量的类型和名字(标志符）。

在c++中，除非带了extern关键字，否则都是变量的定义。extern关键字被发明出来恰恰就是为了将声明和定义区分开。

如果带有extern关键字：

```
extern int x=1;//这是定义
extern int x;//这是声明
```

B：同一个函数可以有多次声明，但只能有一次实现（定义）。

C：在编译各编译单元（cpp）时，编译器会记录每个编译单元可以提供给其他编译单元使用的函数，以及自己调用了但是缺少定义的函数。在链接过程中，比较所有编译单元记录的情况。如果一个函数只定义了而不被调用，它不会被记录为后者，则链接不会出问题。

D：（g++是c++语言的编译器，gcc是c语言的编译器。）

编译过程（不需要函数的定义，只需要函数在头文件中的声明）会把代码编译为【二进制目标文件】（目标文件扩展名是.o)

-c指令是只编译、汇编目标代码，不链接，得到的是目标文件.

链接过程（同时需要函数的定义和声明）则把.o文件转化为可执行文件（扩展名是.exe,也可以不加）

-o指令是用来指定生成的可执行文件名的（如果不指定，默认是a.out)

g++ main.o func1.o func2.o -o main (这个main就是你指定的可执行文件名)

2

A：

Makefile用make指令调用的时候，其编译和链接规则为：

> 1.如果这个工程没有编译过，那么我们的所有Cpp文件都要编译并被链接。
>
> 2.如果这个工程的某几个Cpp文件被修改，那么我们只编译被修改的Cpp文件，并链接目标程序。
>
> 3.如果这个工程的头文件被改变了，那么我们需要编译引用了这几个头文件的Cpp文件，并链接目标程序。

因而只有所有的Cpp文件都被修改的时候才会执行所有任务分别编译生成.o文件的任务。

注：

```
$@--目标文件，  $^--所有的依赖文件，$<--第一个依赖文件。
```

个人建议：考场上如果只让你用makefile生成main文件，那么Makefile里就只写一句：

第一行main:表示你生成的target文件名是main；冒号后面是为了生成该target的前提条件；（也就是所有的cpp文件）

下面-o main表示人工指定目标文件名是main;

```
main: main.cpp a.cpp b.cpp c.cpp d.cpp
    g++ -o main main.cpp
```

B：只能是tab，不能是四个空格

C: C是错的，个人理解是可以在调用函数和变量前的任何地方include?

D：在头文件里声明函数，在源文件里定义函数。如果其他源文件要调用该函数，应该include头文件（函数声明），防止函数重复定义

3

```
z=2*(3+(3+1)*5+1>>1);//替换FUNC和N;
z=2*(24>>1)=24;//注意<<和>>算子的优先级低于加减乘除
```

实际上用含参宏替换时，为了保证正确性，常常要多加括号。

```
#define sqr(x)(x*x)
int s=sqr(3+2)
cout<<s;//3+2*3+2=11,错误
```

```
#define sqr(x)((x)*(x))
int s=sqr(3+2)//25.正确
```

4

int argc,const char* argv[]是命令行参数。int argc是参数个数，argv则是一个char*的数组（指针的数组，每个指针指向一个字符串）。argv[0]是指向第一个参数字符串的指针，...以此类推。

Command Arugument里每出现一个空格，就算是一个新的参数。

如果我的command argument是g++ -o main main.cpp

```
argc=4;
argv[0]=g++;
argv[1]=-o;
argv[2]=main//以此类推
```

特例：如果command argument里出现双引号“”,那么双引号之间的东西会被视为一个参数，无论其中是否有空格。

5

如果函数参数有缺省值，则有缺省值的参数要放在后面。

6

A：inline关键字是“建议”，是否内联由编译器决定；

B：参数类型（包括可以自动类型转换的参数类型，如int和float），以及形参个数（不考虑是否有缺省值），都可以区分函数

C：只有返回值不同的函数无法重载，编译器不知道该调用哪个；

D：内联函数的代码在**编译期**插入在每个调用位置。由此事实得到的推论则是，内联函数的声明和定义都要写在头文件里。因为编译器在编译期就需要得到内联函数的实现，并将它插入在正确的位置。而在编译期，编译器并不会去跨多个源文件去找函数的定义。所以，只能把inline函数的实现写在头文件里，头文件include在调用inline函数的所有cpp文件里，相当于其定义被复制到了每个cpp文件里。

7

A:没有设置权限默认private

B:对的，缺省值无法消除二义性

C:func(P a)的确修改了a的私有成员变量，因而是错的

但是你根本没调用这个函数，这不是这段代码不能编译的原因

8

A：(*this).a就等同于this->a;但如果是静态成员函数，就不能访问该类的任何非静态的成员或是函数了。

C：同一个auto关键字只能推导一个类型，比如

```
auto a=5,b=3,c=2;//这是对的
auto a=3,b=1.8;//这是错的
```

不同类型会导致auto推导失败。

关于auto：

> 1.auto声明的变量必须就地初始化；
>
> 2.auto要在编译期确定类型；
>
> 3.函数参数不能是auto类型；
>
> 4.同一个auto关键字应将变量推导为同一类型：