Lesson 13. Variable Modifiers (变量修饰符) — Auto & Extern

变量修饰符是指用于修改变量的行为或属性的附加关键字或标记

Auto Modifier

Auto means Automatic
Variables declared inside a scope by default are automatic variables
(默认情况下，在作用域内声明的变量是自动变量)
自动变量的生命周期与函数的执行周期相匹配，也就是说，它们在函数被调用时创建，在函数执行结束后销毁，*该变量不会浪费任何内存*

Syntax： auto int some_variable_name;
自C99标准以后，C语言引入了`auto`关键字的新用法，用于显式声明存储类别为自动的局部变量。然而，这种用法几乎不再使用，因为局部变量的默认存储类别就是自动的，所以通常不需要显式地使用`auto`

Take Aways (要点)

- If you won't initialize auto variable, by default it will be initialized with some garbage (random) value

```c
#include <stdio.h>

int main()
{
    auto int var;
    printf("%d", var);
    return 0;
}
```

```
"E:\C Programming & Data St
8
Process returned 0 (0x0)   execution time : 0.663 s
Press any key to continue.
```

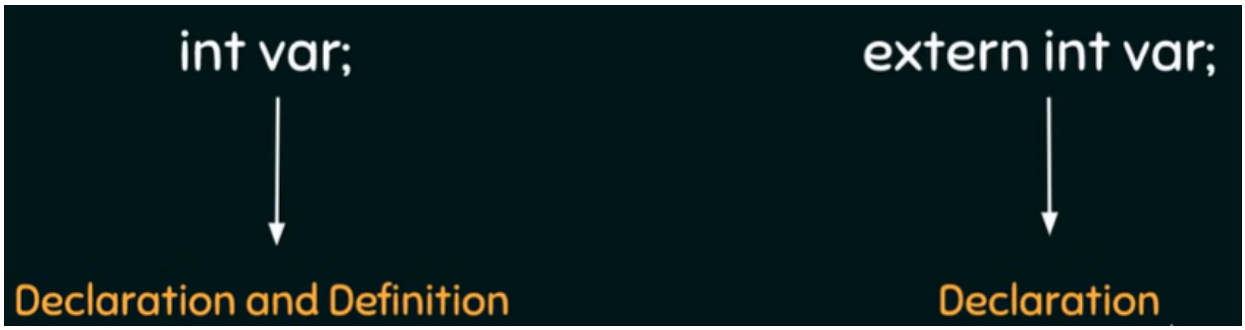
- On the other hand, global variable by default(默认) initialized to 0

```c
#include <stdio.h>

int var;

int main()
{
    printf("%d", var);
    return 0;
}
```

```
"E:\C Programming & Data St
0
Process returned 0 (0x0)   execution time : 0.394 s
Press any key to continue.
```

Extern Modifier

Extern is short name for external

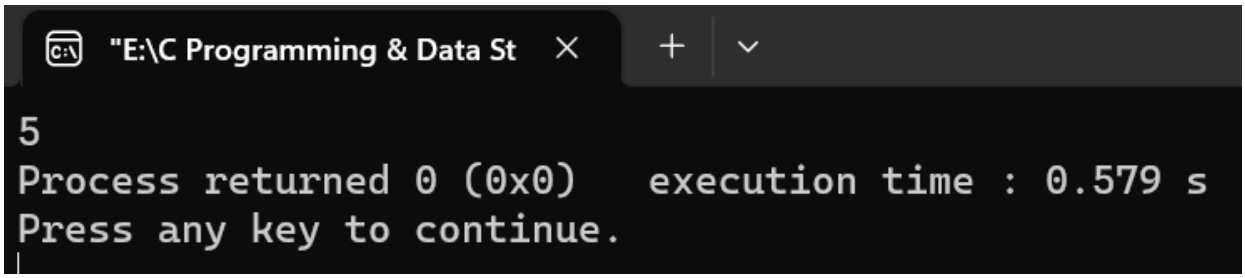


Used when a particular file needs to access a variable from another file

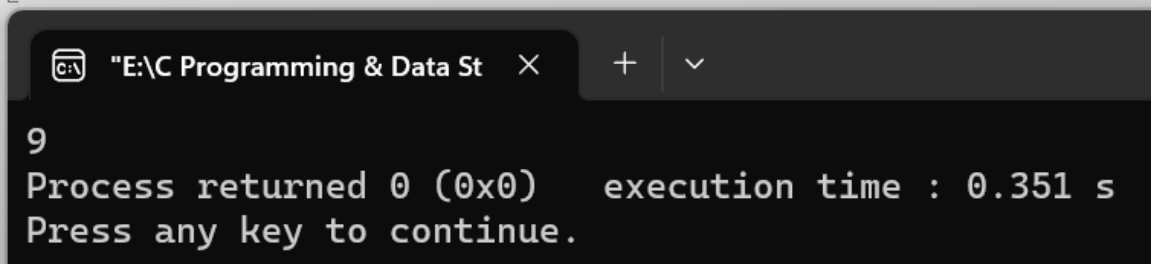
`extern`关键字用于声明一个全局变量或函数，以指示它们实际上是在其他源文件中定义的，而不是在当前源文件中定义的。这允许在不同的源文件之间共享变量或函数，以便它们在整个程序中可用

main.c
```c
#include <stdio.h>

extern int a;

int main()
{
    printf("%d", a);
    return 0;
}
```

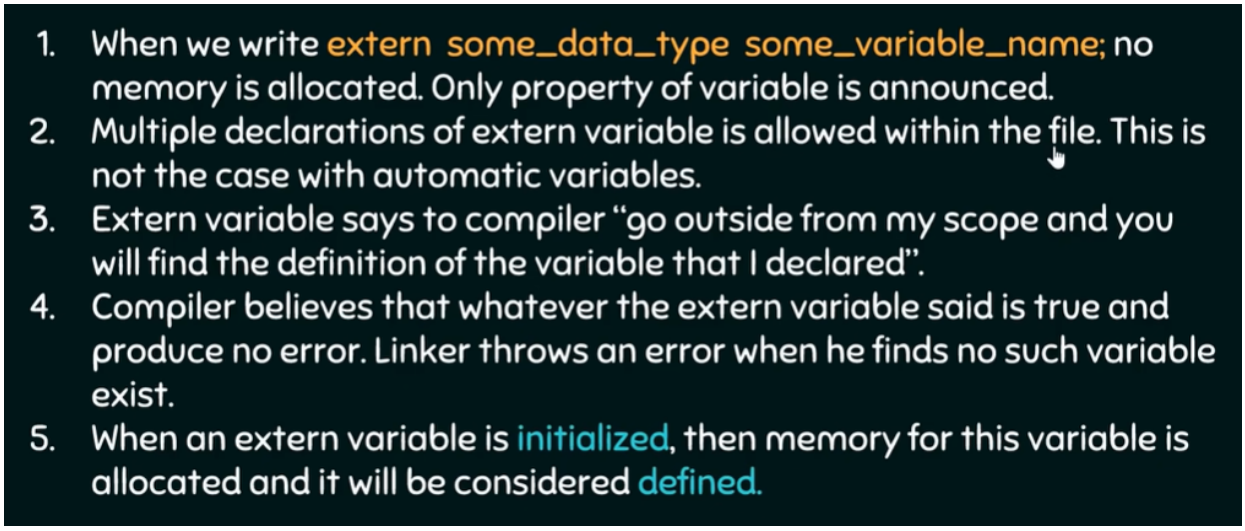
other.c
```c
int a = 5;
```




```c
#include <stdio.h>

int a = 9;

int main()
{
    extern int a;
    printf("%d", a);
    return 0;
}
```



Take Aways



1.当我们写extern some_data_type some_variable_name; 没有分配内存。 仅声明变量的属性。
2.文件内允许多次声明extern变量。 自动变量的情况并非如此。
3.Extern 变量对编译器说：“从我的作用域之外出去，你将找到我声明的变量的定义”。
4.编译器认为外部变量所说的都是正确的，不会产生错误。 当链接器发现不存在这样的变量时，会抛出错误。
5.当一个外部变量被初始化时，该变量的内存被分配，并且它将被视为已定义。