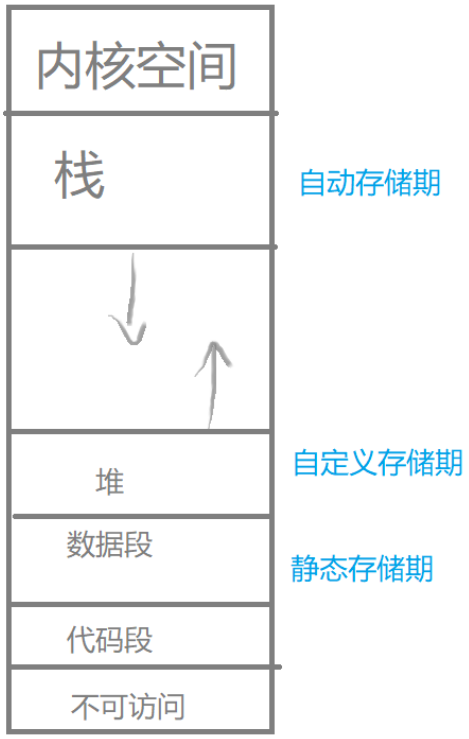
概念：

C语言中， 每一个变量都有一个生命周期，所谓的生命周期指的是某一个变量的内存从申请到释放的过程。申请一个内存相当于某一个变量的诞生，释放掉该内存则是相当于消亡。

变量的生命周期有一下三种形式：

- 自动存储期
- 静态存储期
- 自定义存储期



自动存储期：

在栈空间中分配的变量（内存） ， 统统由系统统一管理，用户不需要担心，因此就称为自动存储期。

有一下几个概念是等价的：

1. 自动化变量： 从存储期的角度来描述变量的存储特性
2. 临时变量： 从存储期的角度来描述变量的存储特性
3. 局部变量： 从作用域的角度来表述变量的空间特定

```
1 int main(int argc , char const * argv[]) // argc argv 自动存储期 / 局部变量
2 {
```

```
 int a ; // 自动存储期 / 局部变量
 static int b ; // 静态存储期 局部变量
 func(a , b);
}

void func(int x , int y) // x, y 自动存储期 / 局部变量
{

 }

```

静态存储期：

在数据段中分配的变量（内存），统统都称为静态存储期，静态存储期的内存，在程序运行之初就已经分配好，不会随着程序的运行发生申请和释放的问题，直到整个程序退出才会释。换句话说他的生命周期与进程一致。

静态存储期：

- 全局变量，static只是影响他的作用域，并不影响他的存储期
- static修饰的局部变量，对于局部变量而言static只是改变了变量的存储期，而没有改变他的作用域。

```
int a = 100 ;
static int b = 250 ; // a b 都属于静态存储期， 只不过b的作用域为本文件

int main()
{
 int k ;

 int a = 350 ;
 static int k = 450 ; // 静态存储期
 }

```

注意：

.bss段 存放的是未初始化的静态变量（静态存储期） 初始值为 0

.data 段 已经初始化的静态变量，初始化语句只会被执行一次

静态数据从进程运行之初已经存在，直到进程退出为止

自定义存储期：

在堆内存中分配的变量（内存），都属于自定义存储期，他的申请与释放完全由用户自己把握。

如何申请：

malloc　calloc　realloc

如何释放：

free

如何清空：

bzero

memset

注意：

malloc 只负责申请空间，并不会清空内存，因此一般使用bzero 清空

calloc 负责申请内存，并会默认清空为 0 .

free 只负责释放，也不会清空 更不会让指针指向空，因此free之后最好让指针指向NULL