**内存分区**

1. 程序运行之前
   1. Text（代码区） 【只读的、共享的】
   2. Data（初始化的静态/全局数据区） 【静态区(static)/全局区/常量区(字符常量、const修饰的){常量区内容时不可修改的}】
   3. Bss（未初始化的…………）
2. 程序运行时
   1. 多出两个区
      1. 栈区（由编译器分配和销毁、适用于数据量你小的）
      2. 堆区（自己分配和自己销毁、适用于数据量大的）
3. 栈区
   1. 自动分配，自动释放
   2. 主要放函数参数、返回值、局部变量
   3. 例子1
      1. 代码

int\* test01(){

int a = 10;

return &a;

}

void test02(){

int \*p = test01();

printf("p = %d\n", \*p);

}

int main()

{

test02();

return 0;

}

* + 1. 上面代码运行会出现错误，为什么
       1. 我们定义的a时局部变量，放在栈区
       2. 那么栈区的特点就是该函数运行结束后所有的内容会清空
       3. 那么，函数运行完，是不是意味着a变量会被回收呀
       4. 那此时我们再返回a的地址，是不是会出错，因为此时a的地址时属于系统，万一有有用的数据，是不是会导致程序出错
  1. 例子2
     1. 代码

char\* test01(){

char a[] = "hello world";

return a;

}

void test02(){

char \*p = NULL;

p = test01();

printf("p = %s\n", \*p);

}

int main()

{

test02();

return 0;

}

* + 1. 我们来进行分析一下
       1. 我们的char a[] 是放在栈区的
       2. 我们的”hello world” 是放在常量区的
       3. 那么我们用图文并茂的方式来讲解一下

