



## § 6. 指针基础 – 画内存图并分析

要求:

- 1、模仿第06模块PDF课件中 (P. 16-19/P. 27-30) 的样式，画出下列每小题每一步执行的内存分配及指向图示，分析为什么得到最后的结果。
  - ★ PDF课件的P. 31（如何同时得到周长和面积）
  - ★ PDF课件的P. 32（为什么无法进行交换）
  - ★ PDF课件的P. 33（为什么会出现错误，导致错误的关键词是哪一句）
- 2、每个语句要画一张内存状态图，每小题都是4张图
  - ★ 第1张初始内存分配图附件已给出
- 3、不允许手写、手写后贴图
- 4、转换为pdf后在“文档作业”中提交（5.16前）



## § 6. 指针基础 – 画内存图并分析

★ PDF课件的P. 31（如何同时得到周长和面积）

```
#include <iostream>
using namespace std;
```

```
#define PI 3.14159
```

```
double SL(double R, double *L)
```

```
{ double S;
```

```
  S = PI*R*R;
```

```
  *L = 2*PI*R;
```

```
  return S;
```

```
}
```

```
int main()
```

```
{ double s, l, r=3;
```

```
  s=SL(r, &l);
```

```
  cout << "s=" << s << endl;
```

```
  cout << "l=" << l << endl;
```

```
}
```

函数执行后同时得到周长及面积

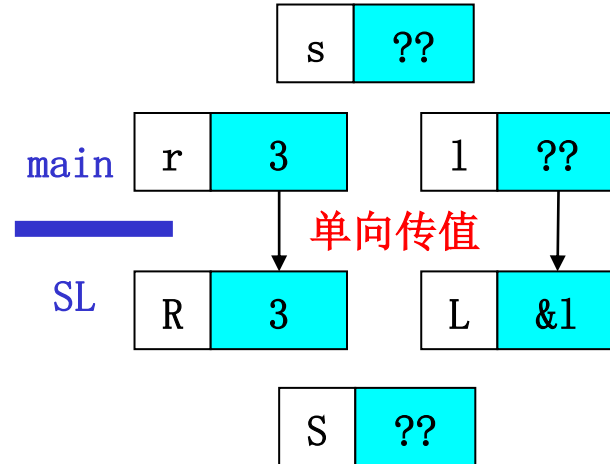
周长：指针变量做形参方式

面积：函数返回值方式

注：函数的return只能带一个返回值!!

s=28.2743

l=18.8495



初始内存分配如图所示  
请自行画出SL中三句话  
执行时内存的变化  
理解最后的输出结果



## § 6. 指针基础 – 画内存图并分析

★ PDF课件的P. 31（如何同时得到周长和面积）

```
#include <iostream>
using namespace std;
```

```
#define PI 3.14159
```

```
double SL(double R, double *L)
```

```
{ double S;
```

```
  S = PI*R*R;
```

```
  *L = 2*PI*R;
```

```
  return S;
```

```
}
```

```
int main()
```

```
{ double s, l, r=3;
```

```
  s=SL(r, &l);
```

```
  cout << "s=" << s << endl;
```

```
  cout << "l=" << l << endl;
```

```
}
```

函数执行后同时得到周长及面积

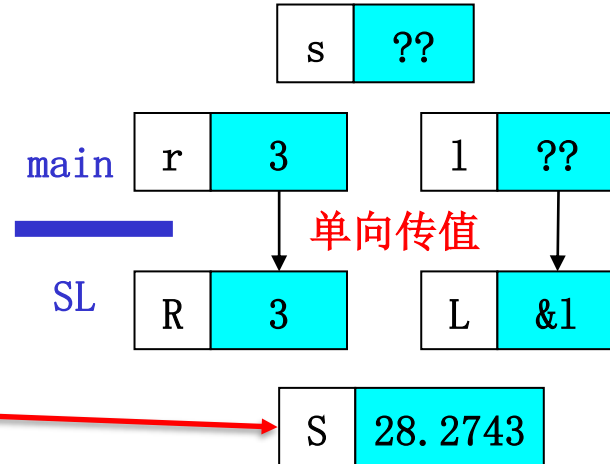
周长：指针变量做形参方式

面积：函数返回值方式

注：函数的return只能带一个返回值!!

s=28.2743

l=18.8495



初始内存分配如图所示  
请自行画出SL中三句话  
执行时内存的变化  
理解最后的输出结果



## § 6. 指针基础 – 画内存图并分析

★ PDF课件的P. 31（如何同时得到周长和面积）

```
#include <iostream>
using namespace std;
```

```
#define PI 3.14159
```

```
double SL(double R, double *L)
{
    double S;
    S = PI*R*R;
    *L = 2*PI*R;
    return S;
}
```

```
int main()
{
    double s, l, r=3;
    s=SL(r, &l);
    cout << "s=" << s << endl;
    cout << "l=" << l << endl;
}
```

函数执行后同时得到周长及面积

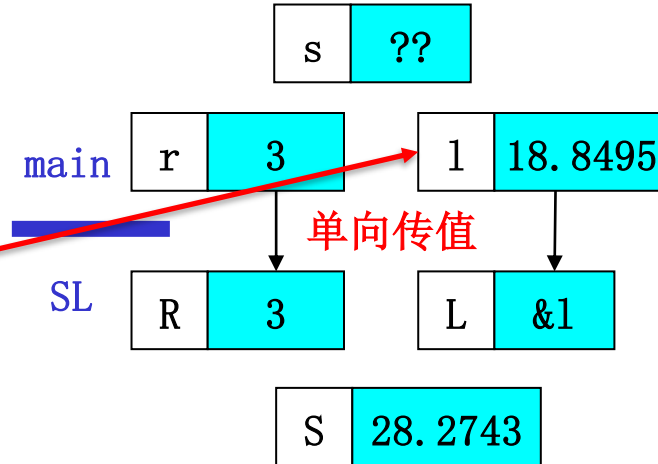
周长：指针变量做形参方式

面积：函数返回值方式

注：函数的return只能带一个返回值!!

s=28.2743

l=18.8495



初始内存分配如图所示  
请自行画出SL中三句话  
执行时内存的变化  
理解最后的输出结果



## § 6. 指针基础 – 画内存图并分析

★ PDF课件的P. 31（如何同时得到周长和面积）

```
#include <iostream>
using namespace std;
```

```
#define PI 3.14159
```

```
double SL(double R, double *L)
```

```
{ double S;
```

```
  S = PI*R*R;
```

```
  *L = 2*PI*R;
```

```
  return S;
```

```
}
```

```
int main()
```

```
{ double s, l, r=3;
```

```
  s=SL(r, &l);
```

```
  cout << "s=" << s << endl;
```

```
  cout << "l=" << l << endl;
```

```
}
```

函数执行后同时得到周长及面积

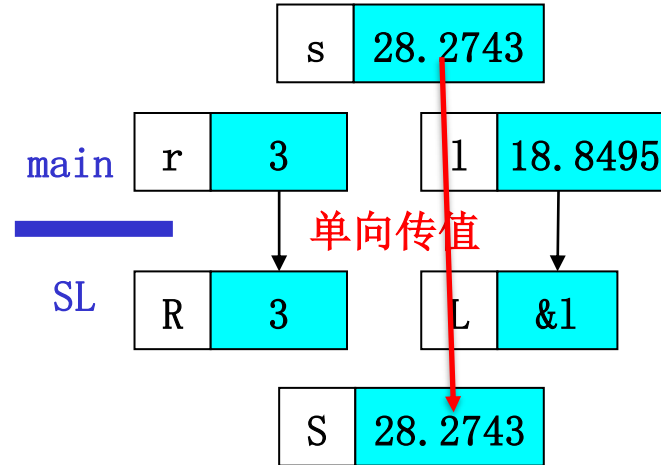
周长：指针变量做形参方式

面积：函数返回值方式

注：函数的return只能带一个返回值!!

s=28.2743

l=18.8495



初始内存分配如图所示  
请自行画出SL中三句话  
执行时内存的变化  
理解最后的输出结果

周长是通过指针变量作为形参，间接访问实参l；  
面积是通过函数的返回值来传递的。

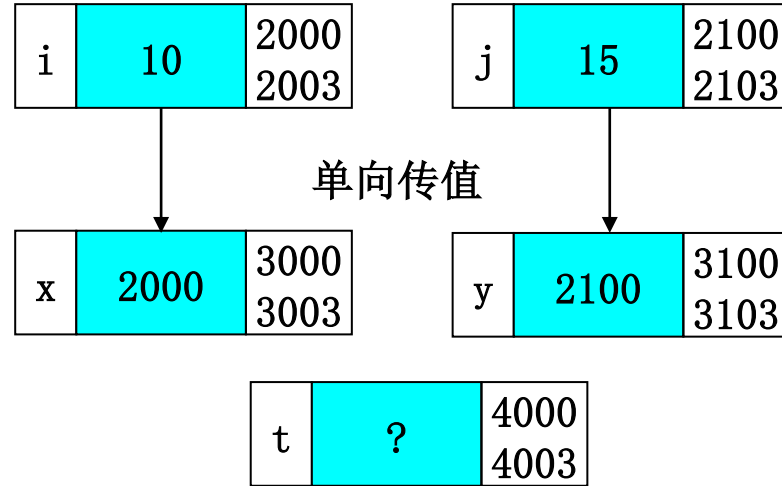


## § 6. 指针基础 – 画内存图并分析

★ PDF课件的P. 32（为什么无法进行交换）

```
void swap(int *x, int *y)
{
    int *t;
    t = x;
    x = y;
    y = t;
}
```

```
int main()
{
    int i=10, j=15;
    cout << "i=" << i << " j=" << j << endl;
    swap(&i, &j);
    cout << "i=" << i << " j=" << j << endl;
}
```



初始内存分配如图所示  
请自行画出swap中三句话  
执行时内存的变化  
理解为什么无法交换



## § 6. 指针基础 – 画内存图并分析

★ PDF课件的P. 32（为什么无法进行交换）

```
void swap(int *x, int *y)
{
    int *t;
    t = x;
    x = y;
    y = t;
}

int main()
{
    int i=10, j=15;
    cout << "i=" << i << " j=" << j << endl;
    swap(&i, &j);
    cout << "i=" << i << " j=" << j << endl;
}
```

单向传值

i	10	2000 2003
---	----	--------------

j	15	2100 2103
---	----	--------------

x	2000	3000 3003
---	------	--------------

y	2100	3100 3103
---	------	--------------

t	2000	4000 4003
---	------	--------------

初始内存分配如图所示  
请自行画出swap中三句话  
执行时内存的变化  
理解为什么无法交换



## § 6. 指针基础 – 画内存图并分析

★ PDF课件的P. 32（为什么无法进行交换）

```
void swap(int *x, int *y)
{
    int *t;
    t = x;
    x = y;
    y = t;
}

int main()
{
    int i=10, j=15;
    cout << "i=" << i << " j=" << j << endl;
    swap(&i, &j);
    cout << "i=" << i << " j=" << j << endl;
}
```

单向传值

i	10	2000 2003
---	----	--------------

j	15	2100 2103
---	----	--------------

x	2100	3000 3003
---	------	--------------

y	2100	3100 3103
---	------	--------------

t	2000	4000 4003
---	------	--------------

初始内存分配如图所示  
请自行画出swap中三句话  
执行时内存的变化  
理解为什么无法交换





## § 6. 指针基础 – 画内存图并分析

★ PDF课件的P. 32（为什么无法进行交换）

```
void swap(int *x, int *y)
{
    int *t;
    t = x;
    x = y;
    y = t;
}

int main()
{
    int i=10, j=15;
    cout << "i=" << i << " j=" << j << endl;
    swap(&i, &j);
    cout << "i=" << i << " j=" << j << endl;
}
```

初始内存分配如图所示  
请自行画出swap中三句话  
执行时内存的变化  
理解为什么无法交换

因为函数参数是单向传值，  
且交换的是地址（函数中的形参），并没有交换地址所指向的变量，无法回传到i和j。

单向传值

i	10	2000	2003
---	----	------	------

j	15	2100	2103
---	----	------	------

x	2100	3000	3003
---	------	------	------

y	2000	3100	3103
---	------	------	------

t	2000	4000	4003
---	------	------	------



# § 6. 指针基础 – 画内存图并分析

★ PDF课件的P. 33（为什么会出现错误，导致错误的关键语句是哪一句）

```
void swap(int *x, int *y)
{
    int *t;
    *t = *x;
    *x = *y;
    *y = *t;
}

int main()
{
    int i=10, j=15;
    cout << "i=" << i << " j=" << j << endl;
    swap(&i, &j);
    cout << "i=" << i << " j=" << j << endl;
}
```

**VS编译报错**  
-使用了未初始化的局部变量t  
其它编译器可能可以运行  
初始内存分配如图所示，请自行画出  
swap中三句话执行时内存的变化，理解为什么出现严重错误

另1: 哪句是错误的键?  
另2: int \*t 改为 int tt, \*t;  
t = &tt;  
为什么就正确了?

i	10	2000
		2003

j	15	2100
		2103

单向传值

x	2000	3000
		3003

y	2100	3100
		3103

t	(假设5000)	4000
		4003

	?	5000
		5003

提示: 5000-5003系统是否分配给了程序?

i=10 j=15

i=15 j=10

或 死机或其它非正常现象



## § 6. 指针基础 – 画内存图并分析

★ PDF课件的P. 33（为什么会出现错误，导致错误的关键语句是哪一句）

```
void swap(int *x, int *y)
{
    int *t;
    *t = *x;
    *x = *y;
    *y = *t;
}

int main()
{
    int i=10, j=15;
    cout << "i=" << i << " j=" << j << endl;
    swap(&i, &j);
    cout << "i=" << i << " j=" << j << endl;
}
```

**VS编译报错**  
-使用了未初始化的局部变量t  
其它编译器可能可以运行  
初始内存分配如图所示，请自行画出swap中三句话执行时内存的变化，理解为什么出现严重错误  
另1：哪句是错误的键？  
另2：int \*t 改为 int tt, \*t;  
t = &tt;  
为什么就正确了？

单向传值

i	10	2000	2003
j	15	2100	2103
x	2000	3000	3003
y	2100	3100	3103
t	(假设5000)	4000	4003
	10	5000	5003

提示：5000-5003系统是否分配给了程序？

i=10 j=15  
i=15 j=10  
或 死机或其它非正常现象



## § 6. 指针基础 – 画内存图并分析

★ PDF课件的P. 33（为什么会出现错误，导致错误的关键语句是哪一句）

```
void swap(int *x, int *y)
{
    int *t;
    *t = *x;
    *x = *y;
    *y = *t;
}

int main()
{
    int i=10, j=15;
    cout << "i=" << i << " j=" << j << endl;
    swap(&i, &j);
    cout << "i=" << i << " j=" << j << endl;
}
```

**VS编译报错**  
-使用了未初始化的局部变量t  
其它编译器可能可以运行  
初始内存分配如图所示，请自行画出  
swap中三句话执行时内存的变化，理解为什么出现严重错误

另1: 哪句是错误的键?  
另2: int \*t 改为 int tt, \*t;  
t = &tt;  
为什么就正确了?

i	15	2000
		2003

j	15	2100
		2103

单向传值

x	2000	3000
		3003

y	2100	3100
		3103

t	(假设5000)	4000
		4003

	10	5000
		5003

提示: 5000-5003系统是否分配给了程序?

i=10 j=15

i=15 j=10

或 死机或其它非正常现象



## § 6. 指针基础 – 画内存图并分析

★ PDF课件的P. 33（为什么会出现错误，导致错误的关键语句是哪一句）

```
void swap(int *x, int *y)
{
    int *t;
    *t = *x;
    *x = *y;
    *y = *t;
}

int main()
{
    int i=10, j=15;
    cout << "i=" << i << " j=" << j << endl;
    swap(&i, &j);
    cout << "i=" << i << " j=" << j << endl;
}
```

**VS编译报错**  
-使用了未初始化的局部变量t  
其它编译器可能可以运行  
初始内存分配如图所示，请自行画出swap中三句话执行时内存的变化，理解为什么出现严重错误  
另1: 哪句是错误的关键? `int *t`  
另2: `int *t` 改为 `int tt, *t;`  
      `t = &tt;`  
      为什么就正确了?

单向传值

i	15	2000	2003
---	----	------	------

j	10	2100	2103
---	----	------	------

x	2000	3000	3003
---	------	------	------

y	2100	3100	3103
---	------	------	------

t	(假设5000)	4000	4003
---	----------	------	------

	10	5000	5003
--	----	------	------

提示: 5000-5003系统是否分配给了程序?

`i=10 j=15`  
`i=15 j=10`  
或 死机或其它非正常现象

5000-5003系统没有分配给程序，因为定义指针\*t未附初值，则随机指向，无法正常使用；  
若将`int *t` 改为 `int tt, *t;` `t=&tt;`，则正确是因为定义tt后，使t=&tt，系统已分配4字节给程序，属于合法使用