编译原理与技术

第7章　运行环境

wenshli@bupt.edu.cn

李文生

2024年2月19日 星期一

北京邮电大学计算机学院(国家示范性软件学院)
SCHOOL OF COMPUTER SCIENCE(NATIONAL PILOT SOFTWARE ENGINEERING SCHOOL)BUPT

学习任务

- 作业要求
  - 参数传递机制的应用
  - 程序运行期间的空间组织及控制栈的状态其变化过程

wenshli@bupt.edu.cn

运行环境考题示例

五、（15 分）对于如下的 Pascal 程序，画出程序运行到（1）和（2）点时的控制栈的状态，要求标出各活动记录中的活动名称、控制链和访问链。

```
program main(input, output);
    VAR i:integer; d:integer;
    procedure A(k:real);
        VAR p:char;
        procedure B;
            VAR c:char;
            begin
                …(1)…
            end; {end of B}
        procedure C;
            VAR t:real;
            begin
                …(2)…
            end; {end of C}
        begin
            ……
            B;
            C;
            ……
        end; {end of A}
    begin
        …
        A(d);
        …
    end.{end of main}
```

六、（15 分）有如下 C 语言程序：

```
int m, n;
void QQ(int x, int y)
{
    x=x*m;
    y=y+m;
    x=x-y;
    m=y-x;
}
main()
{
    m=3;
    n=4;
    QQ(m,n);
    printf("m=%d ;   n=%d\n", m, n);
}
```

假定采用下面的参数传递机制，该程序的执行结果分别是什么？

（1）传值调用

（2）引用调用

（3）复制恢复

要求：描述程序执行过程的主要步骤。

五、（15 分）有如下 C 语言程序：

```
#include <stdio.h>
int a, b;

void p(int x, int y, int z) {
    y = y + x;
    a = a * b;
    z = z + 10;
    b++;
}

int main() {
    a = 3;
    b = 2;
    p(a+b, a, b);
    printf("a = %d, b = %d", a, b);
}
```

假定采用下面的参数传递机制，该程序的执行结果分别是什么？

（1） 传值调用

（2） 引用调用

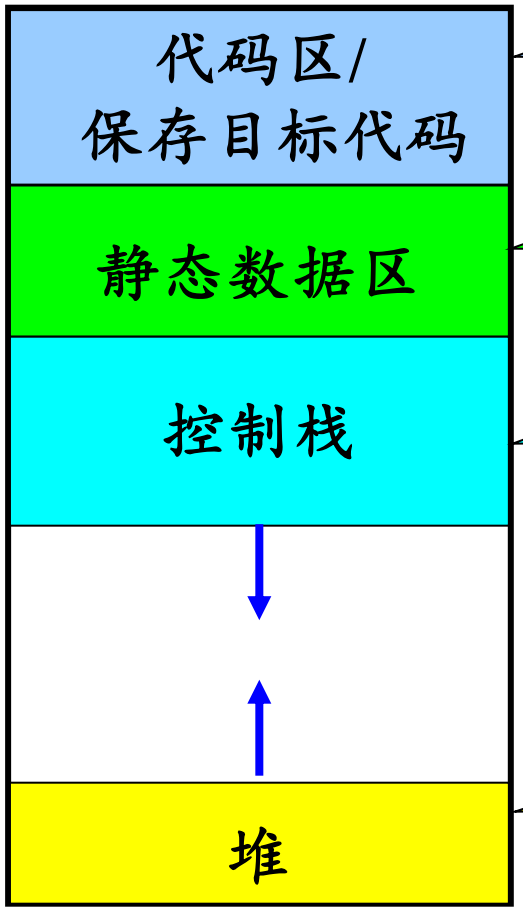
（3） 复制恢复

要求：描述程序执行过程的主要步骤。

wenshli@bupt.edu.cn

程序运行空间的划分

- 编译程序在编译源程序时，向操作系统申请一块内存区域，以便被编译程序在其中运行。



编译时可以确定代码段的长度，
可以放在一个静态确定的区域内

编译时可确定大小的数据对象，放在静态确定的区域内，目标地址可以编入目标代码中

存储过程的运行环境，用于管理过程的活动、保存断点的现场信息，用于返回时的恢复

程序控制下进行的动态存储分配，
分配在堆中，如Pascal中new函数

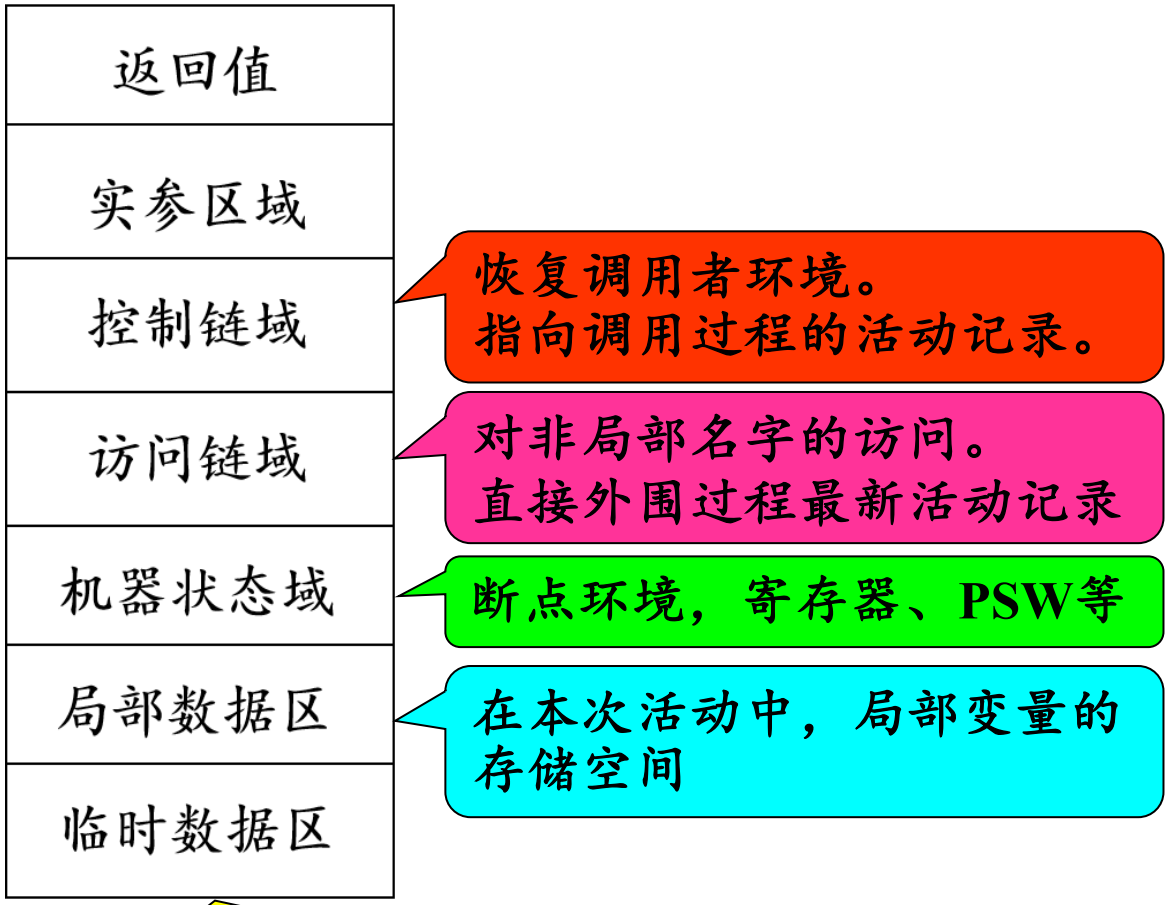
wenshli@bupt.edu.cn

活动记录与控制栈　　活动记录的内容与组织

- 允许过程递归调用的语言（如Pascal、C等），过程块的活动空间不能静态分配。
  - 某段时间内，一个递归过程可能有多个活着的活动。

```
int f(int n) {
    if (n==0) return 1;
    else return n*f(n-1);
}
```

  - 需要为过程的每次活动分配不同的存储空间。
  - 通常采用栈式存储分配策略。
- 活动记录：一个连续的存储块
  - 记录过程在一次执行中所需要的信息。



wenshli@bupt.edu.cn

★考点1: 控制栈和活动记录

有如下C语言程序：

```
include<strio.h>
include<stdlib.h>
int i;
int count;
int fun (int i) {
    count++;
    if (i==1) return 1;
    else return i*fun(i-1);
    };
int main() {
    i=5;
    count=0;
    printf( “%d”,  fun(i) );
}
```

问题：

画出该程序运行时存储空间的组织示意图，并画出该程序运行过程中，当count=3时，控制栈中活动记录的示意图。

要求：

画出每个活动记录中的控制链域和访问链域，并给出参数域的值

wenshli@bupt.edu.cn

★考点1: 控制栈和活动记录（答案）

有如下C语言程序：

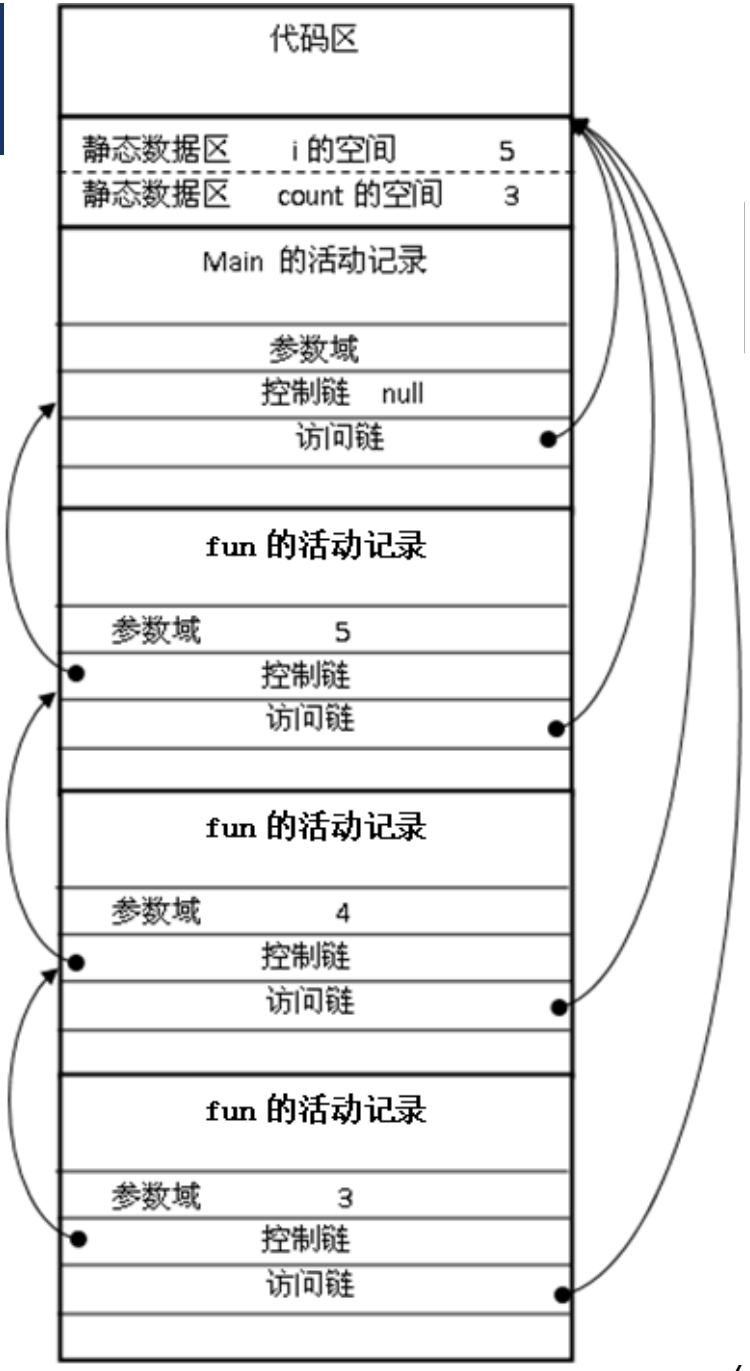
```
include<strio.h>
include<stdlib.h>
int i;
int count;
int fun (int i) {
    count++;
    if (i==1) return 1;
    else return i*fun(i-1);
    };
int main() {
    i=5;
    count=0;
    printf( “%d”,  fun(i) );
}
```

问题：

画出该程序运行时存储空间的组织示意图，并画出该程序运行过程中，当count=3时，控制栈中活动记录的示意图。

要求：

画出每个活动记录中的控制链域和访问链域，并给出参数域的值



wenshli@bupt.edu.cn

★考点2: 参数传递机制

```
int i;
int B[2];
void P(int m)
 {
   i=0;  m=m+10;  B[i]=10;
   i=1;  m=m+10;  B[i]=10;
 }
main()
 {
   B[0]=10;    B[1]=20;
   i=0;    P(B[i]);
   printf("B[0]=d%, B[1]=d%", B[0], B[1]);
 }
```

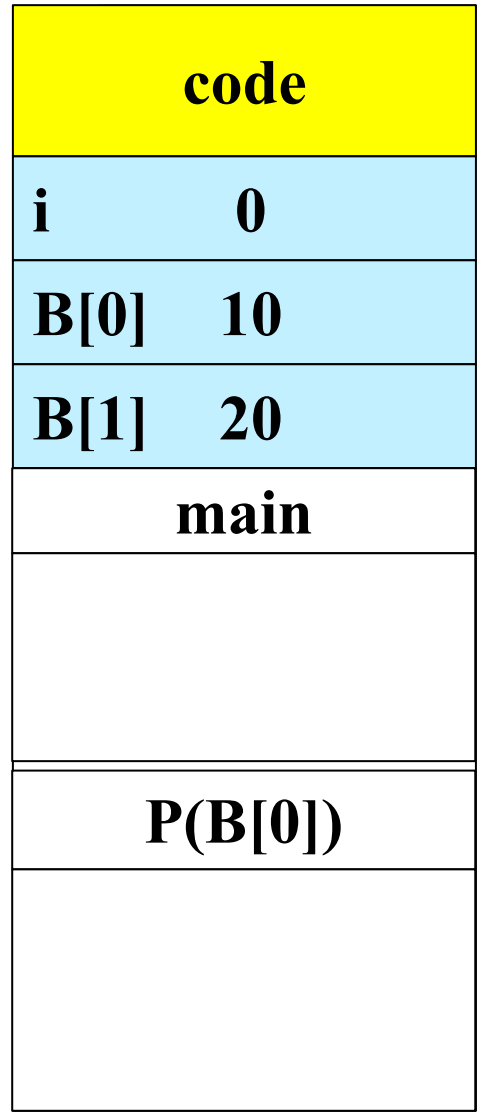
- 传值调用
- 引用调用
- 复制恢复
- 传名调用

wenshli@bupt.edu.cn

★解答：传值调用

```
int i;
int B[2];
void P(int m)
  {
    i=0;  m=m+10;  B[i]=10;
    i=1;  m=m+10;  B[i]=10;
  }
main()
  {
    B[0]=10;    B[1]=20;
    i=0;    P(B[i]);
    printf("B[0]=d%, B[1]=d%", B[0], B[1]);
  }
```

语句	全局变量			形参
	i	B[0]	B[1]	m
	0	10	20	
P(B[i])				
i=0				
m=m+10				
B[i]=10				
i=1				
m=m+10				
B[i]=10				
返回后				
printf(…)				

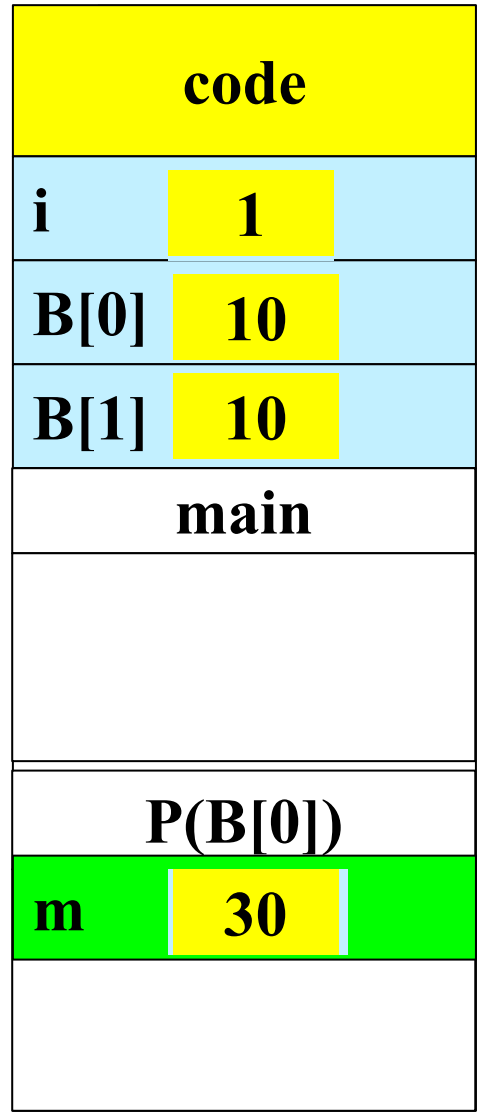


wenshli@bupt.edu.cn

★解答：传值调用（答案）

```
int i;
int B[2];
void P(int m)
  {
    i=0;  m=m+10;  B[i]=10;
    i=1;  m=m+10;  B[i]=10;
  }
main()
  {
    B[0]=10;    B[1]=20;
    i=0;    P(B[i]);
    printf("B[0]=d%, B[1]=d%", B[0], B[1]);
  }
```

语句	全局变量			形参
	i	B[0]	B[1]	m
	0	10	20	
P(B[i])				10
i=0	0			
m=m+10				20
B[i]=10		10		
i=1	1			
m=m+10				30
B[i]=10			10	
返回后	1	10	10	
printf(…)		10	10	

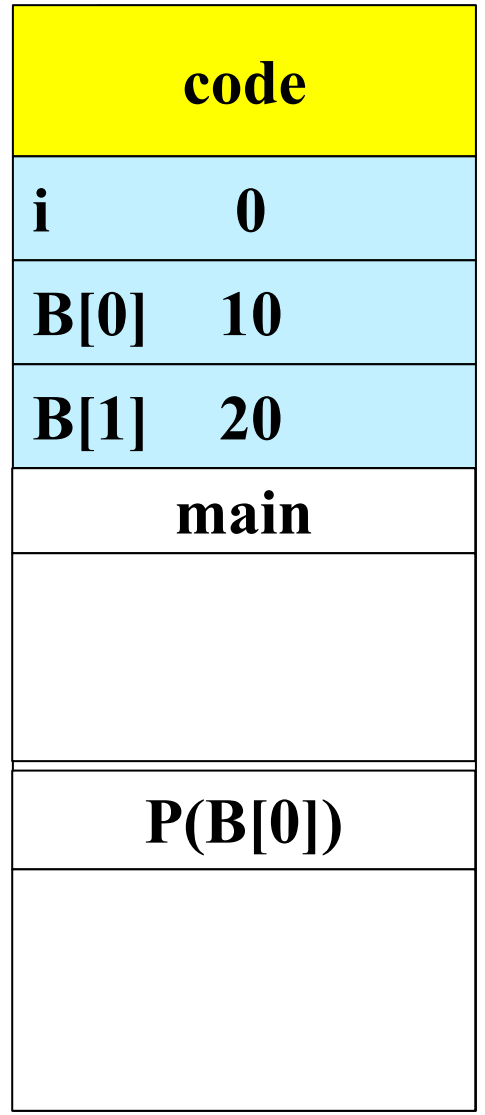


wenshli@bupt.edu.cn

★解答：引用调用

```
int i;
int B[2];
void P(int m)
  {
    i=0;  m=m+10;  B[i]=10;
    i=1;  m=m+10;  B[i]=10;
  }
main()
  {
    B[0]=10;    B[1]=20;
    i=0;    P(B[i]);
    printf("B[0]=d%, B[1]=d%", B[0], B[1]);
  }
```

语句	全局变量			形参
	i	B[0]	B[1]	m
	0	10	20	
P(B[i])				
i=0				
m=m+10				
B[i]=10				
i=1				
m=m+10				
B[i]=10				
返回后				
printf(…)				

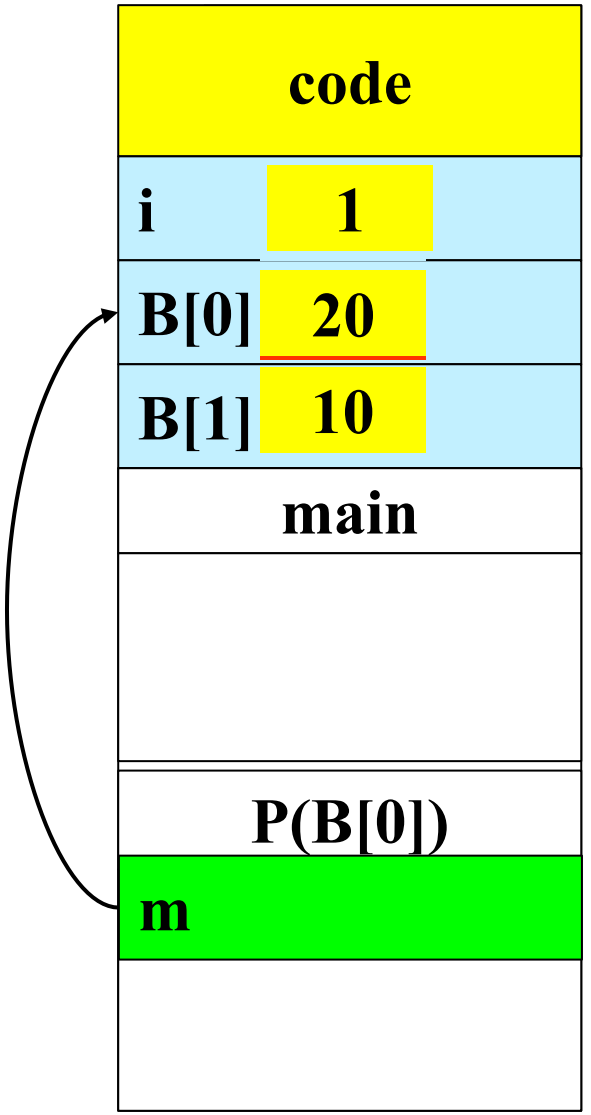


wenshli@bupt.edu.cn

★解答：引用调用（答案）

```
int i;
int B[2];
void P(int m)
  {
    i=0;  m=m+10;  B[i]=10;
    i=1;  m=m+10;  B[i]=10;
  }
main()
  {
    B[0]=10;    B[1]=20;
    i=0;    P(B[i]);
    printf("B[0]=d%, B[1]=d%", B[0], B[1]);
  }
```

语句	全局变量			形参
	i	B[0]	B[1]	m
	0	10	20	
P(B[i])				B[0]
i=0	0			
m=m+10		20		
B[i]=10		10		
i=1	1			
m=m+10		20		
B[i]=10			10	
返回后	1	20	10	
printf(…)		20	10	



wenshli@bupt.edu.cn

★解答：复制恢复

```
int i;
int B[2];
void P(int m)
  {
    i=0;  m=m+10;  B[i]=10;
    i=1;  m=m+10;  B[i]=10;
  }
main()
  {
    B[0]=10;    B[1]=20;
    i=0;    P(B[i]);
    printf("B[0]=d%, B[1]=d%", B[0], B[1]);
  }
```

语句	全局变量			形参
	i	B[0]	B[1]	m
	0	10	20	
P(B[i])				
i=0				
m=m+10				
B[i]=10				
i=1				
m=m+10				
B[i]=10				
返回后				
printf(…)				

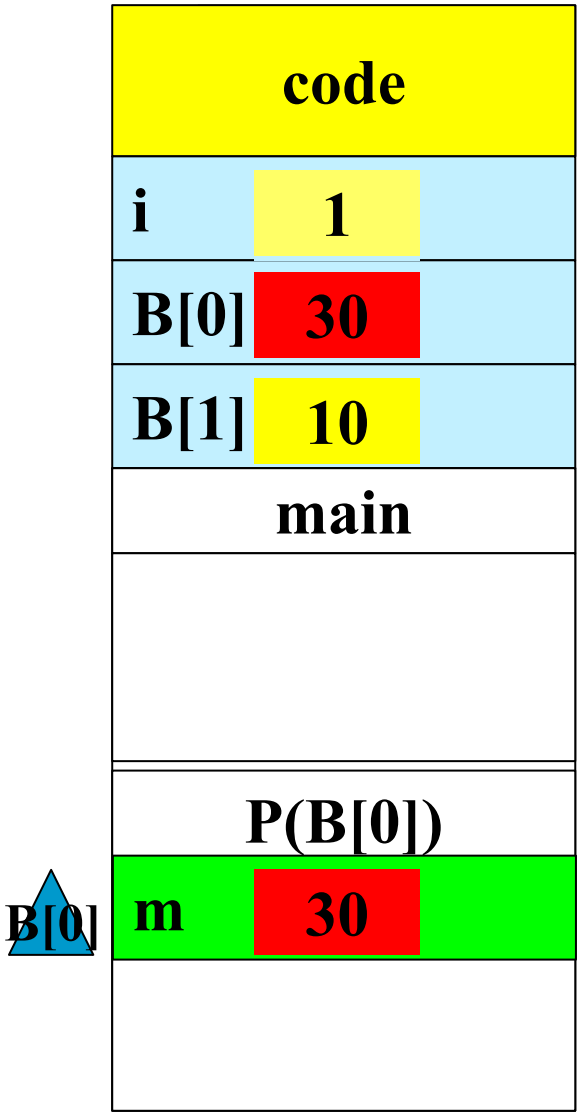
code
i 0
B[0] 10
B[1] 20
main
P(B[0])

wenshli@bupt.edu.cn

★解答：复制恢复（答案）

```
int i;
int B[2];
void P(int m)
  {
    i=0;  m=m+10;  B[i]=10;
    i=1;  m=m+10;  B[i]=10;
  }
main()
  {
    B[0]=10;    B[1]=20;
    i=0;    P(B[i]);
    printf("B[0]=d%, B[1]=d%", B[0], B[1]);
  }
```

语句	全局变量			形参
	i	B[0]	B[1]	m
	0	10	20	
P(B[i])				10 -- B[0]
i=0	0			
m=m+10				20
B[i]=10		10		
i=1	1			
m=m+10				30
B[i]=10			10	
返回后	1	30	10	
printf(…)		30	10	



wenshli@bupt.edu.cn

★解答：传名调用

```
int i;
int B[2];
void P(int m)
 {
   i=0;  m=m+10;  B[i]=10;
   i=1;  m=m+10;  B[i]=10;
 }
main()
 {
   B[0]=10;    B[1]=20;
   i=0;    P(B[i]);
   printf("B[0]=d%, B[1]=d%", B[0], B[1]);
 }
```

wenshli@bupt.edu.cn

★解答：传名调用

```
int i;
int B[2];
void P(int m)
 {
   i=0; m=m+10; B[i]=10;
   i=1; m=m+10; B[i]=10;
 }
main()
 {
   B[0]=10;   B[1]=20;
   i=0;   P(B[i]);
   printf("B[0]=d%, B[1]=d%", B[0], B[1]);
 }
```

```
i=0;
B[i]=B[i]+10;
B[i]=10;
i=1;
B[i]=B[i]+10;
B[i]=10;
```

wenshli@bupt.edu.cn

★解答：传名调用（答案）

```
int i;
int B[2];
void P(int m)
 {
   i=0; m=m+10; B[i]=10;
   i=1; m=m+10; B[i]=10;
 }
main()
 {
   B[0]=10;   B[1]=20;
   i=0;   P(B[i]);
   printf("B[0]=d%, B[1]=d%", B[0], B[1]);
 }
```

```
i=0;
B[i]=B[i]+10;
B[i]=10;
i=1;
B[i]=B[i]+10;
B[i]=10;
```

```
B[0]=20
B[0]=10
B[1]=30
B[1]=10
```

执行结果：

B[0]=10, B[1]=10

wenshli@bupt.edu.cn

★运行环境考题示例--参数传递机制

有如下C语言程序：

```
int i;
int b[4];
void Q(int x; int y) {
    i=1;
    x=x+2;    b[i]=15;
    y=y+3;    b[i]=20;
}
main() {
    for (i=0;i<4;i++) b[i]=i;
    i=1;
    Q(b[i], b[i+1]);
    for (i=0; i<4; i++)
        printf("b[%d]=%d ", i, b[i]);
}
```

假定采用下面的参数传递机制，该程序的执行结果分别是什么？

(1) 传值调用

(2) 引用调用

(3) 复制恢复

要求：描述程序执行过程的主要步骤。

wenshli@bupt.edu.cn

★解答：(1)传值调用

过程：

	全局变量					形参	
	i	b[0]	b[1]	b[2]	b[3]	x	y
	1	0	1	2	3		
Call Q(b[i], b[i+1])						1	2
i=1	1						
x=x+2						3	
b[i]=15			15				
y=y+3							5
b[i]=20			20				
返回							
结果	1	0	20	2	3		

结果为：b[0]=0　b[1]=20　b[2]=2　b[3]=3

wenshli@bupt.edu.cn

★解答：(2)引用调用

过程：

	全局变量					形参	
	i	b[0]	b[1]	b[2]	b[3]	x	y
	1	0	1	2	3		
Call Q(b[i], b[i+1])						b[1]	b[2]
i=1	1						
x=x+2			3				
b[i]=15			15				
y=y+3				5			
b[i]=20			20				
返回							
结果	1	0	20	5	3		

结果为：b[0]=0　b[1]=20　b[2]=5　b[3]=3

wenshli@bupt.edu.cn

★解答：(3)复制恢复

过程：

	全局变量					形参	
	i	b[0]	b[1]	b[2]	b[3]	x	y
	1	0	1	2	3		
Call Q(b[i], b[i+1])						1 / b[1]	2 / b[2]
i=1	1						
x=x+2						3	
b[i]=15			15				
y=y+3							5
b[i]=20			20				
返回			3	5			
结果	1	0	3	5	3		

结果为：b[0]=0　b[1]=3　b[2]=5　b[3]=3

wenshli@bupt.edu.cn