题目

2019年1月13日 星期日　　22:03

一　lambda演算

用核心 λ 演算证明(20 分):

[1]　1+1 = 2　　[2] zerop 0　=　T

其中:

T = λ x. λ y.x　　//逻辑真值

F = λ x. λ y.y　　//逻辑假值

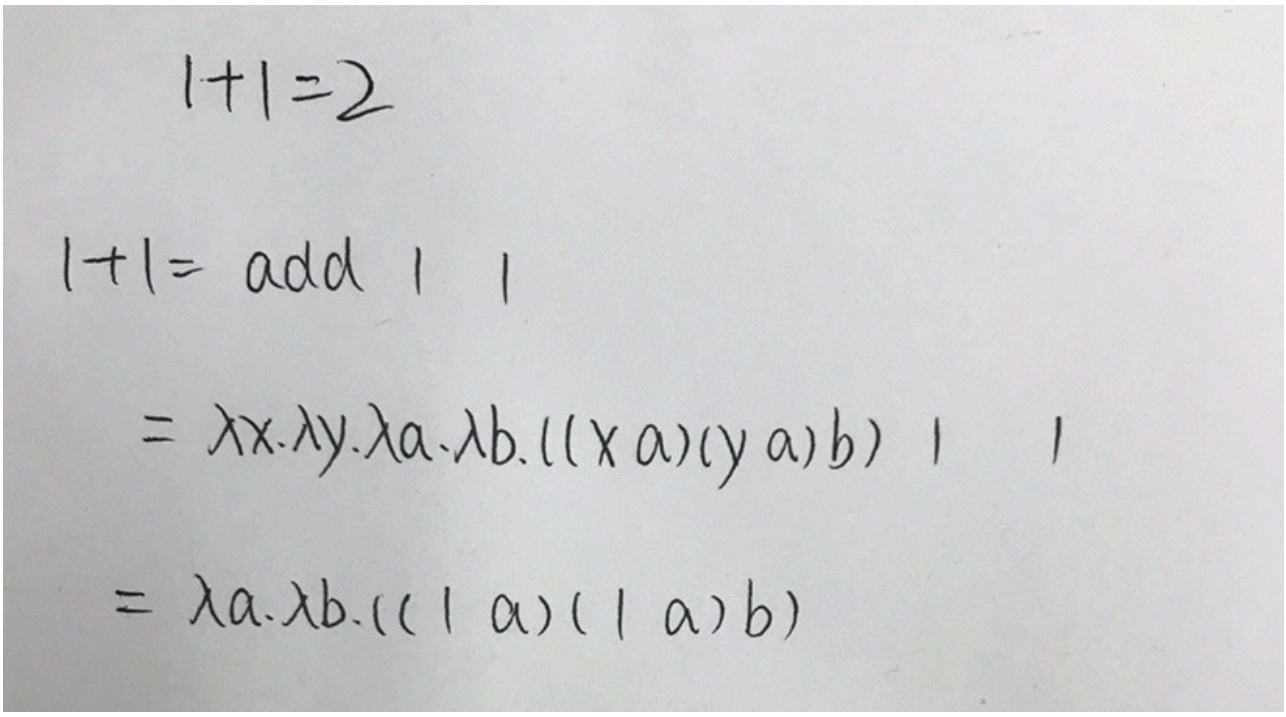
0 = λ x. λ y.y　　//数 0

1 = λ x. λ y.x y　　//数 1

2 = λ x. λ y.x(x y)　　//数 2

zerop = λ n.n(λ x.F)T　　//判零函数

+ = add = λ x. λ y. λ a. λ b.((x a)(y a)b)　　//整数加



$$= \lambda a.\lambda b.((\lambda x_1.\lambda y_1.x_1 y_1\ a)(\lambda x_2.\lambda y_2.x_2 y_2\ a)\ b)$$

$$= \lambda a.\lambda b.((\lambda x_1.\lambda y_1.x_1 y_1\ a)(a\ b)$$

$$= \lambda a.\lambda b.a(a\ b) = 2$$

$$\text{Zerop } 0 = \lambda n.n(\lambda x.F)T\ 0$$

$$= 0(\lambda x.F)T$$

$$= (\lambda x_1.\lambda y_1.y_1)(\lambda w.\lambda x_2.\lambda y_2.y_2)(\lambda x_3.\lambda y_3.x_3)$$

$$= \lambda y_1.y_1(\lambda x_3.\lambda y_3.x_3)$$

$$= \lambda x_3.\lambda y_3.x_3 = T$$

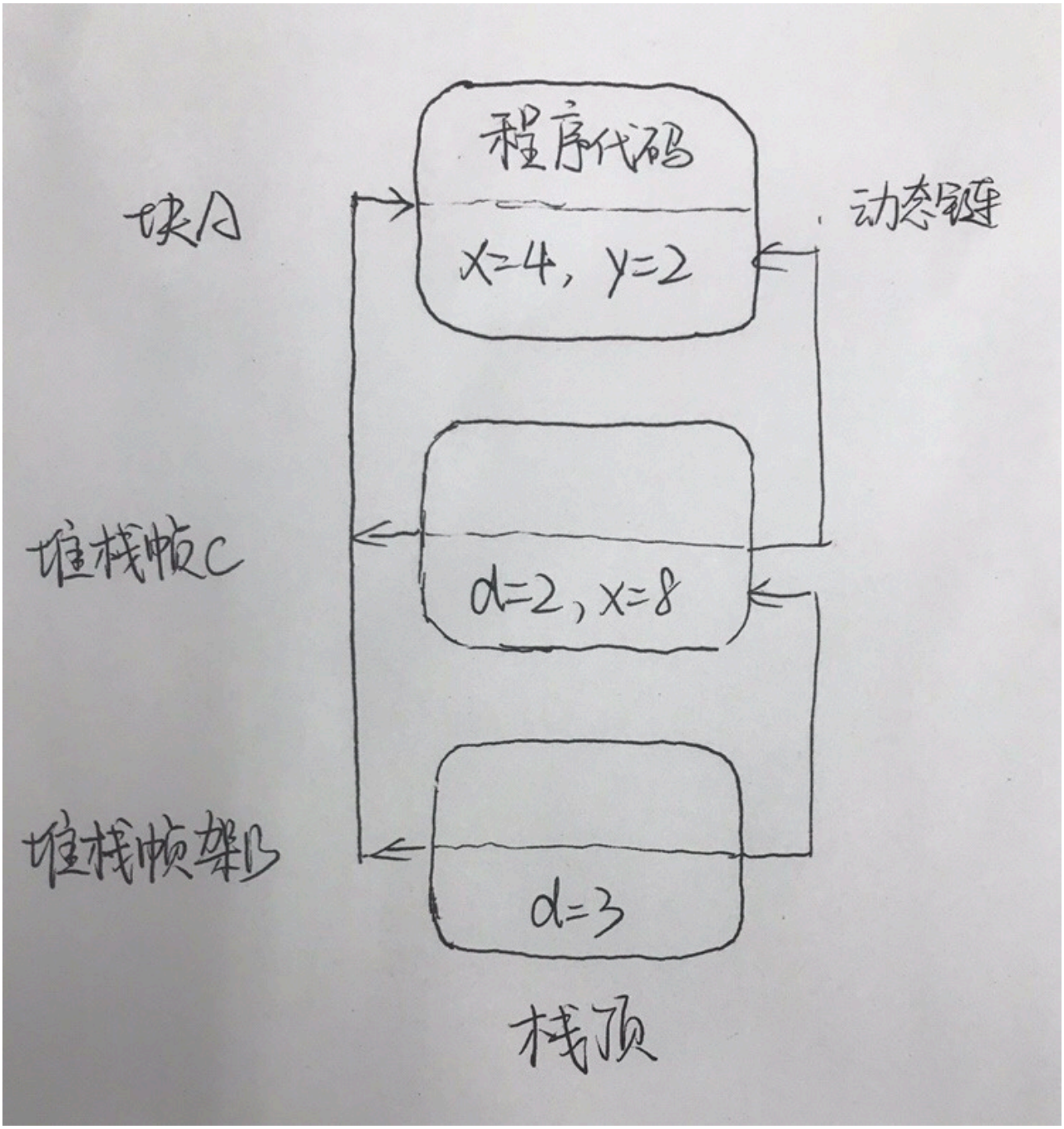
二 词法作用域 动态作用域

请按照词法作用域和动态作用域两种方式给出下面程序执行后的输出打印结果，画出程序执行过程中栈式存储管理的状况图，并基于该图说明词法作用域和动态作用域两种作用域方式的不同点。

```
PROGRAM A;
        VAR x,  y:Integer;
        FUNCTION B(d:Integer): Integer;
            BEGIN B:=x+d+y   END;
        FUNCTION C(d:Integer):Real;
            VAR x:Real;
        BEGIN   x:= 8;   C:=x+B(d+1)   END;
BEGIN   x:= 4;  y := 2;
        Writeln (C(y));
END
```

词法作用域
输出结果：17

动态作用域
输出结果：21



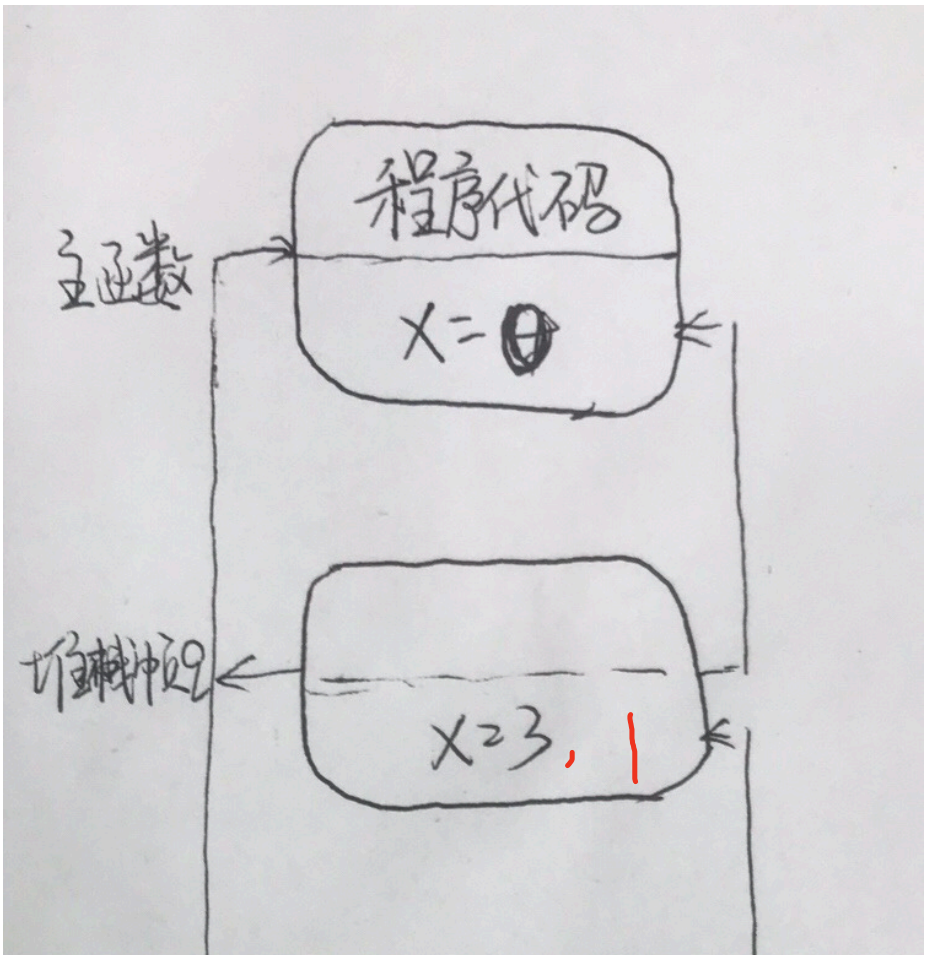
二、请按照词法作用域和动态作用域两种方式给出下面程序执行后的输出打印结果，画出程序在执行 13 行对 r 调用后和 14 行对 p 调用后的栈式存储管理状况图。(20 分)

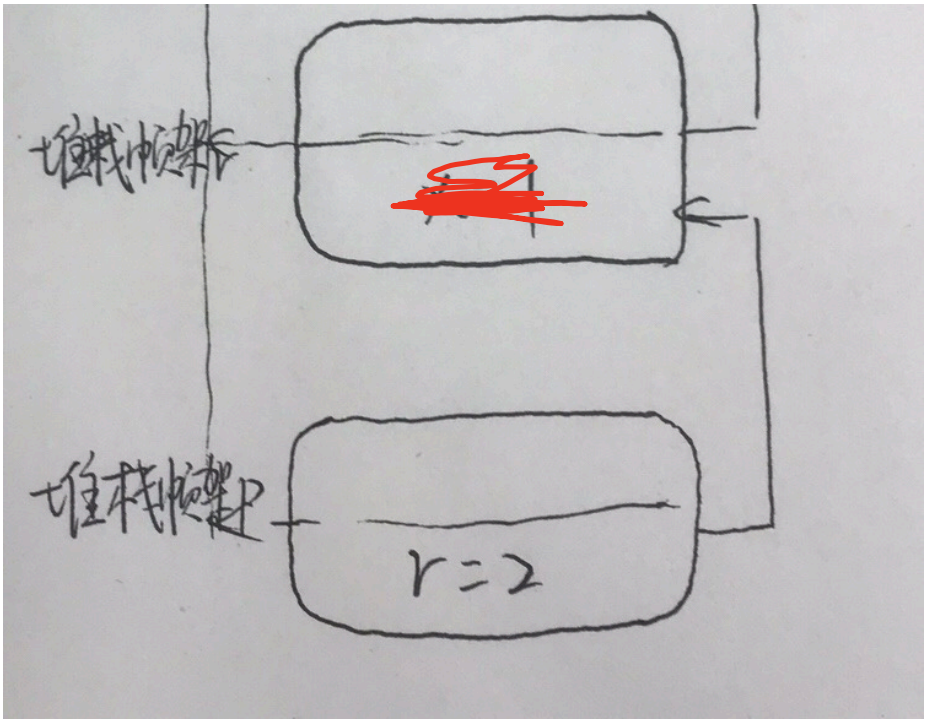
```
(1)  int    x ;
(2)  void    p ( void )
(3)    {    double r = 2;
```

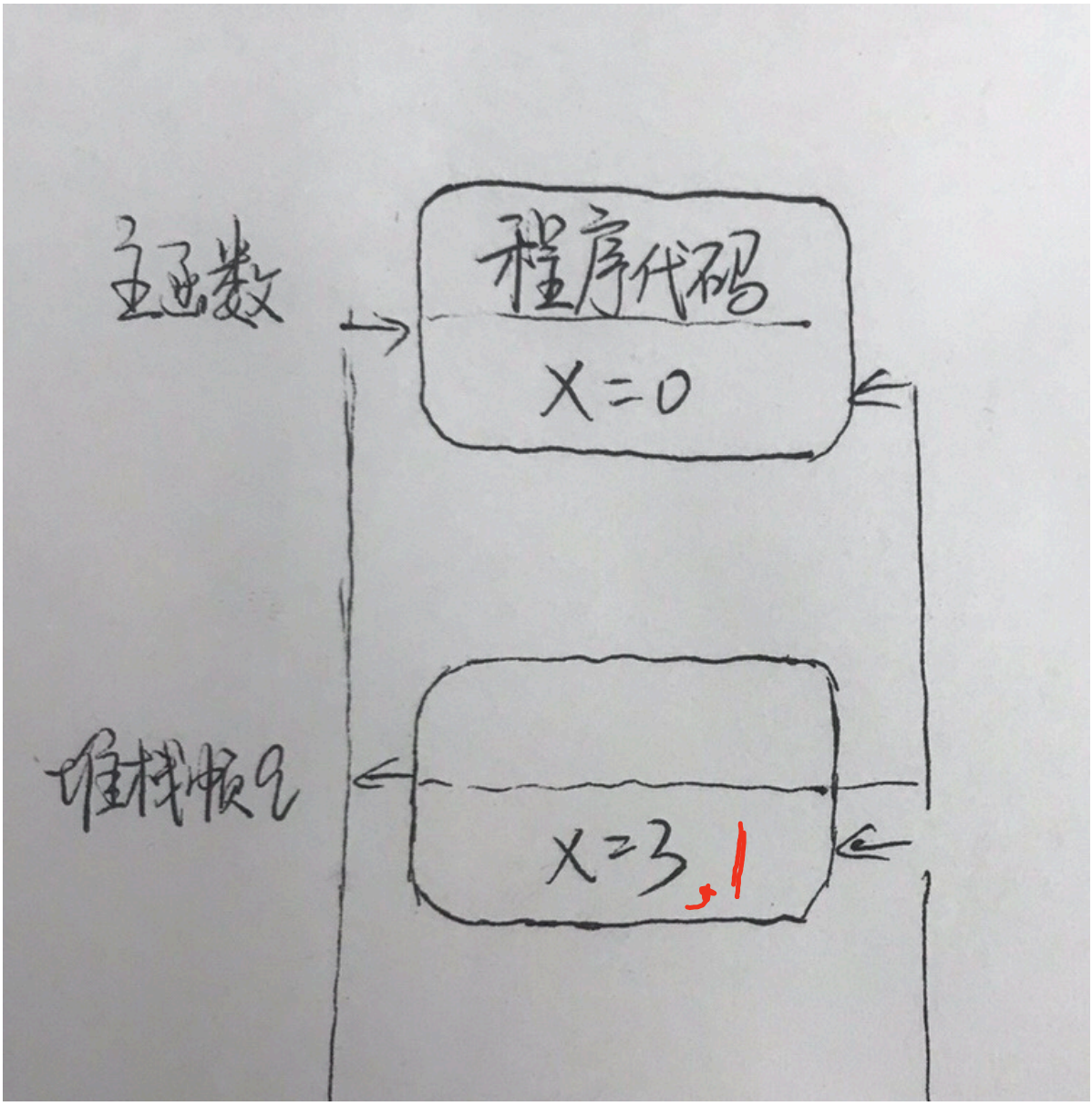
```
(4)         printf ( “%g\n” , r);
(5)         printf( “%d\n” ,x);
(6)    }
(7)  void    r ( void )
(8)    {    x = 1;
(9)         p ( );
(10)  }
(11)  void q ( void )
(12)  {    double x = 3;
(13)        r ( ) ;
(14)        p ( );
(15)  }
(16)  main ( )
(17)  {    p ( ) ;
(18)        q ( );
(19)        return 0;
(20)  }
```

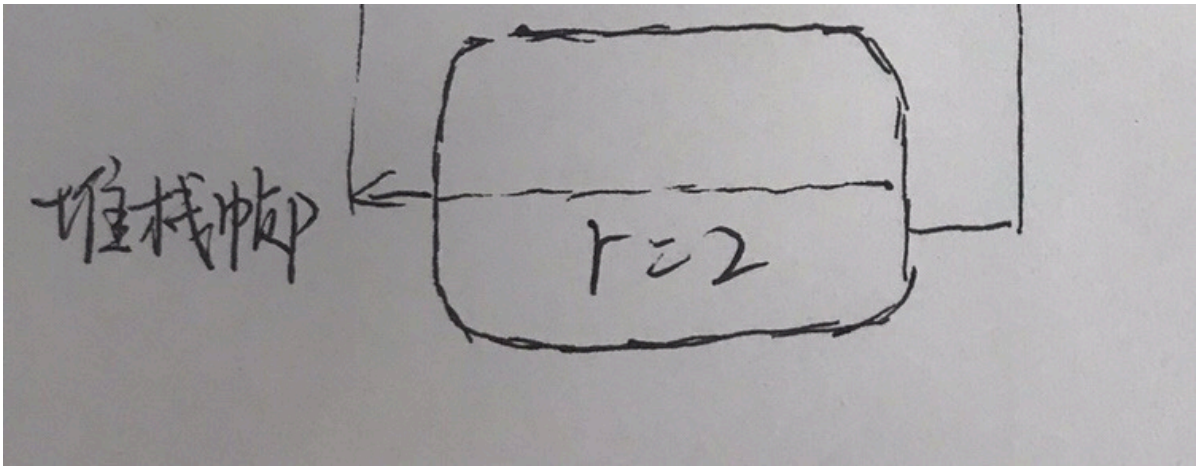
词法作用域输出结果：2 0 2 1 2 1

动态作用域输出结果：2 0 2 1 2 1




r=2


主函数
程序代码
X=0
X=3.1



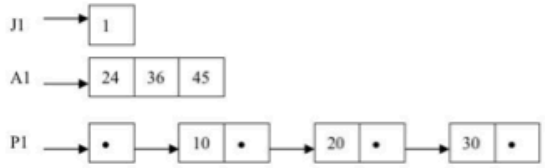
三 参数机制

请论述过程调用的传值、传名和引用三种参数机制的不同点，并说明分别基于三种参数机制调用下述语句的不同执行输出结果以及内存单元状况变化图。

test(J1,A1[J1],P1↑.next);其中过程 test 定义如下

```
PROCEDURE test(J2,A2:Integer;P2:list)
BEGIN
        J2 : = J2+1;
        Writeln(J2,A2, P2↑.value);
        J2 : = J2+1;
        A2 : =J2+ P2↑.value;
        P2 : = P2↑.next;
    Writeln(J2,A2,P2↑.value)
END.
```

变量 J1,A1 和 P1 的当前状态如下：



过程调用中的传值、传名和引用三种参数机制的不同点

传值：把实参值复制到形参上，一般实参运行后不会改变。只能返回单个的函数结果值。

传名：如果在过程中修改了形参的值，就按结合的变元的名字（地址）找出变元值进行修改。在传明过程中虚实结合时将程序体中所有形参出现的地方均以实参变元名置换。

引用：引用传递的是存储对象，实参必须是变量名或能得出地址的表达式。

执行结果：

执行结果：

传值：2 24 20 | 3 23 30

传名：2 36 20 | 3 23 30

引用：2 24 20 | 3 23 30

二、假设下述程序中函数 p 调用可按传值调用、传名调用、引用调用和值返回调用四种参数传递机制进行，请分别按上述四种参数机制说明下述程序的执行输出结果以及描述在函数 p 调用期间的内存单元状况图。(24 分)

```
int   i,j ;
int   a [ 3 ];
void p ( int x , int y )
{   x ++ ;
    i ++ ;
    y ++ ; }
main ( )
{   a [ 0 ] = 0 ;
    a [ 1 ] = 1 ;
    a [ 2 ] = 1 ;
    i = 0 ;
    j = 1 ;
    p ( a [ i ] , a [ j ] ) ;
    printf ( "%d\n" , a [ 0 ] ) ;
    printf ( "%d\n" , a [ 1 ] ) ;
    printf ( "%d\n" , a [ 2 ] ) ;
    return 0 ; }
```

传值：0 1 1

传名：1 2 1

引用：1 2 1

值返回引用：

四 面向对象

请针对下面的一段 Java 代码片断中，1、列出说明存在各种面向类型和面向复合对象的操作，2、说明下述代码构造了哪些新类型，3、程序代码中是否存在错误，为什么；若程序修改正确后，主程序输出的结果是什么？

```
import java.io.*;
public class PassTest{
```

```
    float ptValue;
    public static void main(String args[]) {
        PassTest pt=new PassTest();
        char[] myChars={''a'', ''b'', ''c''};
        String s2 = new String(myChars) ;
        int val=s2.length();
        short b=val;
        String stringArray[ ];
        String stringArray = new String[3];
        stringArray[0]= new String("The ");
        stringArray[1]= new String("Value ");
        stringArray[2]= new String("is ");
        System.out.println(stringArray +val);
        pt.changeInt(val);
        System.out.println(stringArray +val);
        pt.ptValue=b+1;
        System.out.println(stringArray+pt.ptValue);
        pt.changeObjValue(pt);
        System.out.println(stringArray+pt.ptValue);
    }
public void changeInt(int value){
    value=55;
}
public void changeObjValue(PassTest ref){
    ref.ptValue=99;
}
```

1. 哪些操作

   面向类型的操作：赋值、拼接、类型转换

   面向复合对象的操作：构造、赋值、数组索引
2. q
3. 构造了哪些新类型

   PassTest{float: ptValue}

   String[]

4. 代码存在哪些错误

   short b = (short)val;

   stringArray = new String[3];

   System.out.println(stringArray[0] + stringArray[1] + stringArray[2] + val);

5. 程序输出的结果是什么

   The Value is 3

   The Value is 3

The Value is 4
The Vlaue is 99

五 并发编程

三、进程同步的一个典型例子是资源分配问题。若一个系统有三台打印机，那么最多只能有三个进程同时使用打印机，请按照第 14 章 14.1.2 小节的监控器机制编写相应的打印机资源分配的监控器模块，并简单说明客户进程如何使用监控器机制实现上述打印机资源的分配与同步的。(20 分)

```
monitor PRINTER::
    var num = 0;
    var printer_available : cond;//当num > 0时，printer_available为真

    proc get_printer() is:
        while num = 0 do wait (printer_available) end;
        if num > 0 then num = num - 1 endif;
    end;

    proc finish_print() is :
        num = num + 1;
        if num = 1 then signal (printer_available) endif;
    end;
End PRINTER
```

三、试论述信号灯、条件临界区和监控器三种并发机制的基本原理以及三者异同。(20 分)

信号灯、信号量原理、优缺

为了处理忙等待的低级、设计上的低效

信号量是一个非负整值变量s。在其上定义了两个操作P,V(取自荷兰语字头,即wait（等待）和signal（示信）。V操作发信号指示一个事件可以出现,P操作延迟所在进程直至某个事件已经出现

优缺:

信号灯理论则为进程交互的同步与互斥的研究打下了基础

信号灯是语句级，直接作为程序设计语言层次偏低

条件临界区原理、优缺

条件临界区(Condition Critical Region简称CCR)将共享变量显式地置于叫做资源的区域内。每个进程在自己的进程体内指明要访问的条件临界区，而同一临界区可出现在不同进程之中(谁进谁用)

优缺：

条件临界区最主要的优点是概念清晰：

- 无需辅助标志和变量即可描述共享变量的任何进程交互
- 程序编译时即可保证互斥
- 一个进程创建一个条件不需顾及其它条件是否与此条件有关
- 易于程序正确性证明
- 体现了共享数据传递的方便

它的致命缺点是低效(和信号灯相比)。此外：

- 进程和共享变量耦合太紧
- 临界区利写不利读，一多了就太散，因而也难修改

监控器原理、优缺

把分散在整个程序中的**region**语句进一步集中成为一个模块叫做监控器(**monitor**)

二、请采用 Java 语言编写基于贪睡理发师模型的客户服务排队程序 (35 分)

```
import java.util.concurrent.Semaphore;

public class SleepBarber {
	public static void main(String[] args) {
		Semaphore signal = new Semaphore(10);//初始座位数为10
		Semaphore sleep = new Semaphore(0);//初始为睡觉状态
		Thread barber = new Thread(new Barber(signal, sleep));

		barber.start();
		try {
			Thread.sleep(1000);
			while (true) {
				Thread customer = new Thread(new Customer(signal, sleep));
				customer.start();
				Thread.sleep(100);
			}
		} catch (Exception e) {
			e.printStackTrace();
		}
	}
```

```
public static class Customer implements Runnable {
        Semaphore signal;
        Semaphore sleep;

        public Customer(Semaphore signal, Semaphore sleep) {
                this.signal = signal;
                this.sleep = sleep;
        }

        public void run() {
                synchronized (this) {
                        if (signal.tryAcquire()) {//尝试P座位，如果有就进去，没有就
                        离开
                                System.out.println("新增一名顾客，" +
                                signal.availablePermits());
                                if (signal.availablePermits() == 9) {
                                        sleep.release();
                                }
                        } else {
                                System.out.println("座椅不够，客户离开");
                        }
                }
        }
}

public static class Barber implements Runnable {
        Semaphore signal;
        Semaphore sleep;

        public void run() {
                while (true) {
                        synchronized (this) {
                                try {
                                        sleep.acquire();//等待sleep
                                } catch (Exception e) {
                                        e.printStackTrace();
                                }
                                while (signal.availablePermits() > 0) {
                                        try {
                                                Thread.sleep(50);
```

```
                                    } catch (Exception e) {
                                            e.printStackTrace();
                                    }
                                    System.out.println("一名顾客理完发");
                                    if (signal.availablePermits() < 10) {
                                            signal.release();//V操作
                                    }
                            }
                    }
            }
        }

        public Barber(Semaphore signal, Semaphore sleep) {
            this.signal = signal;
            this.sleep = sleep;
        }
    }
}
```

七 选择题

二、判断题（5 道，20 分）

1.1 在 Java 语言中，指针类型不能用于分配基本类型对象，使用动态分配函数 NEW 的返回值是基本数据类型对象，而不是它的引用。

1.2 Pascal 语言提供的集合类型，是采用压缩的 Boolean 数组实现的，集合是枚举的特例。

1.3 在 C 语言语句 typedef struct cc { int a , b ; } newtype ; 中， typedef 和 struct 都不是类型构造子。

1.4 在 Pascal 语言中存在如下语句：

```
type Stack = record
    top : integer;
    store : array (1..100) of stackitems;
end;
```

上述语句中涉及到了 Pascal 记录、数组的类型构造子和复合对象选择子。

1.5 函数是表达式的抽象，过程是语句集的抽象，而类属则是声明集的抽象。

三、判断题（5 道，15 分）

1.6 Pascal 语言是由 N.Wirth 在 ALGOL 语言基础上提出的结构化过程式语言。

1.7 词法内聚准则是指一个模块(或函数、 过程)只用自己声明的局部数据做一件事情。

1.8 函数是命令集的抽象，过程是声明集的抽象，而类属则是表达式的抽象。

1.9 在下述 C 语言代码中，赋值 x 违反了类型兼容性准则，该条赋值语句语法错误：

```
int  x,y;
double  z;
x = double(y)+z;
```

1.10 一个程序设计语言在存储机制中不采用堆式或栈式存储管理方式，就可以消除悬挂引用的问题。

四、选择题（10 道，30 分）

2.1 世界上产生第一个高级程序设计语言是 Fortran 语言，出现的年代为____。

A. 1951 年 B. 1954 年 C.1956 年 D. 1958 年

2.2 一个程序的静态结构，直观上应与动态的计算结构一致，即表示结构与逻辑结构统一，这是 E.W.Dijkstra 提出的著名准则____。

A.频度准则 B.语法一致 C.结构一致 D.安全准则

2.3 程序设计语言的正交性设计准则是指：___

A. 程序中逻辑相关的代码表示上应相邻。

B. 每种语言特征都由独立的机制实现， 即与其他特征的实现机制无关。

C. 看上去相似的特征， 其语法定义应一致。

D. 若某表达陈述不止一次， 则应抽取因子使其以递归形式表达。

2.4 在 C 语言中不是头等程序对象的值是____。

A.整数 B.集合 C.记录 D.指针

2.5 在 C 语言中，模块和包的表现形式是_____

A. 分别编译的文件集

B. 函数或过程模块

C. 类的定义

D. 程序包

2.6 下面关于束定、声明、定义的概念描述哪一句是错误的___。

A. 把名字和存储对象联系起来称之为束定，声明则是产生了事实上的束定。

B. 定义是一种特殊的声明，它为程序对象的名字提供了完整的束定信息；

C. 定义只能一次，声明可以多次；

D. 一个程序对象可以有多个名字，而一个名字只能束定一个程序对象。

2.7 请指出下面____组表达式是有副作用的表达式；

A. let val s = (a+b+c)*0.5
in sqrt(s*(s-a)*(s-b)*(s-c))
end

B. (c= getchar())==' ')

C. if x>y then x else y

D. (if exp the sin else cos)(x)

2.8 当 x=0, y=1 时，在___语言中，表达式 if x>0 then y/x>1 else false 会求值失败；

A.C 语言 B.Ada 语言 C.Pascal 语言 D.ML 语言

2.9 在 ADA 语言中，在下述的声明条件下，_____语句是不合法的。

```
type K is INTEGER;
subtype J is K range 1..100;
type L is new K range 1..100;
a,b: K;  c: J; d:L;
A: a := b+c;
B: a := b+d;
C: c:= (J)d;
D: b:= c+a;
```

2.10 不是面向对象程序设计语言基本特征的是 机制

A. 封装 B.继承 C.多态 D. 类属