

希赛网, 专注于软考、PMP、通信考试的专业 IT 知识库和在线教育平台。希赛网在线题库, 提供历年考试真题、模拟试题、章节练习、知识点练习、错题本练习等在线做题服务, 更有能力评估报告, 让你告别盲目做题, 针对性地攻破自己的薄弱点, 更高效的备考。

希赛网官网: <http://www.educity.cn/>

希赛网软件水平考试网: <http://www.educity.cn/rk/>

希赛网在线题库: <http://www.educity.cn/tiku/>

2011 上半年程序员案例分析真题答案与解析: <http://www.educity.cn/tiku/tp1581.html>

## 2011 年上半年程序员考试下午真题 (参考答案)

•     (1)    

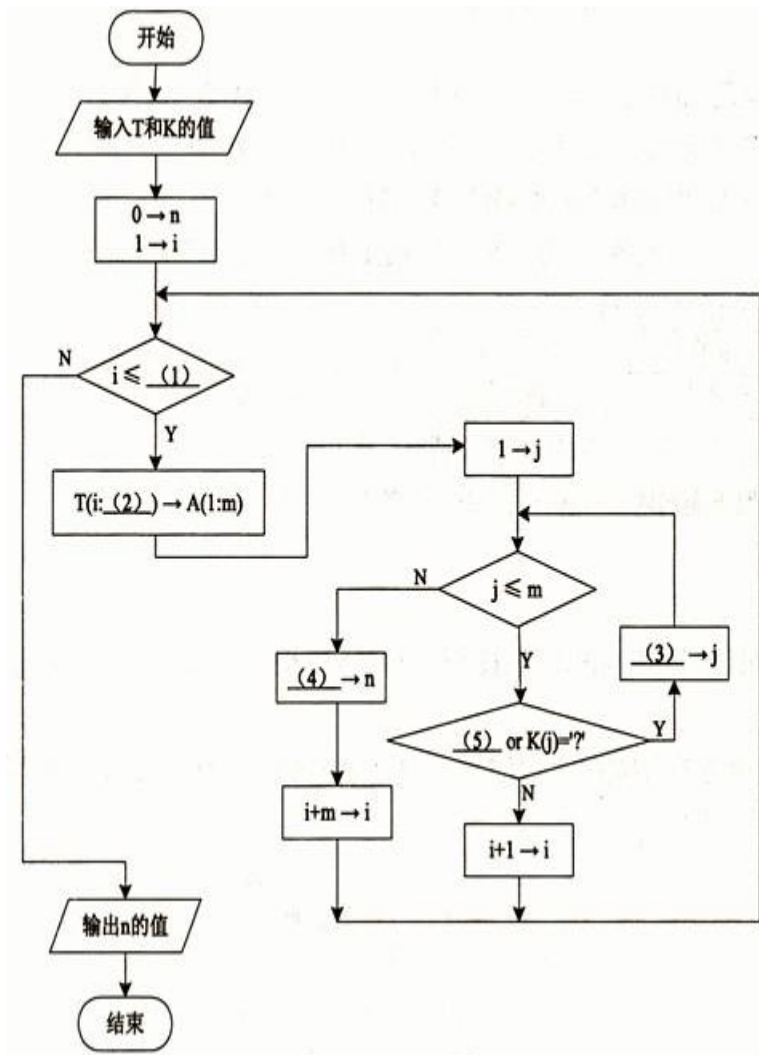
阅读以下说明和流程图, 填补流程图中的空缺(1)~(5), 将解答填入答题纸的对应栏内。

**【说明】**

下面的流程图可在正文字符串  $T(1:L)$  中计算关键词字符串  $K(1:m)$  出现的次数 (用  $n$  表示)。其中,  $L$  为字符串  $T$  的长度,  $m$  为字符串  $K$  的长度 ( $m < L$ )。为便于模糊查找, 关键词中的字符“?”可以匹配任意一个字符。

在该流程图中, 先从  $T$  中取出长度为  $m$  的子串存入  $A$  中, 再将  $A$  与  $K$  进行逐个字符的比较 (其中,  $K$  可以包含字符“?”)。注意: 从正文字符串中取出的关键词字符串不允许交叉。例如, “aaaaaa”中有 3 个关键词字符串“aa”。

**【流程图】**



• (2)

阅读以下说明和 C 函数，回答问题 1 至问题 3，将解答写在答题纸的对应栏内。

**【说明 1】**

函数 `substring(const char str[], int index, int length)` 的功能是求出字符串 `str` 中指定序号 `index` 开始且长度为 `length` 的子串，并返回所取出的子串。以字符串“China today”为例，其第一个字符“C”的序号为 1(而其在字符数组 `str` 中的下标为 0)，从序号 5 开始且长度为 3 的子串为“at”。

**【C 函数 1】**

行号	代码
1	char *substring(const char str[], int index, int length)
2	{
3	char *tptr = 0;
4	int k;
5	/*参数错误, 返回空指针*/
6	if (index < 1    length < 0    index + length - 1 > strlen(str)) return 0;
7	tptr = (char *)malloc((length + 1) * sizeof(char));
8	/*申请子串所需的存储区域*/
9	if (tptr = 0) return 0; /*内存申请失败, 返回空指针*/
10	for(k=0; k < length; k++)
11	tptr[k] = str[index + k - 1];
12	tptr[length - 1] = '\0'; /*字符串置结束标志*/
13	return tptr;
14	}

**【问题 1】 (6 分)**

函数 substring 中有两处错误, 请指出这些错误所在代码的行号, 并在不增加和删除代码行的情况下进行修改, 写出修改正确后的完整代码行 (有注释时, 注释可省略)。

出错行号	修改正确后的完整代码行

**【说明 2】**

在 C 函数 2 中, reverse(unsigned int n) 的功能是求出并返回 n 的反序数。例如, 1234 的反序数是 4321, 其求解过程如下:

- (1) 由 1234 除以 10 得到商 123 和余数 4, 0 乘以 10 再加上 4 得到 4;
- (2) 由 123 除以 10 得到商 12 和余数 3, 4 乘以 10 再加上 3 得到 43;
- (3) 由 12 除以 10 得到商 1 和余数 2, 43 乘以 10 再加上 2 得到 432;
- (4) 由 1 除以 10 得到商 0 和余数 1, 432 乘以 10 再加上 1 得到 4321。

**【C 函数 2】**

```

unsigned int reverse(unsigned int n)
{
    unsigned int result=0;
    while( (1) ){
        result= result *10+ n%10,
        n= (2) ;
    }
    return result,

```

}

**【问题 2】(6 分)**

请根据说明 2，填充 C 函数 2 中的空缺(1)和(2)。

**【问题 3】(3 分)**

用 567, 1234, 56781234, 62354879643 分别作为实参调用函数 reverse，对应的返回值分别为 765, 4321, 43218765, 1357400630。请说明以 62354879643 作为实参调用函数 reverse 时返回结果出错的原因。

( )

阅读以下说明和 C 函数，回答问题 1 至问题 3，将解答写在答题纸的对应栏内。

**【说明 1】**

函数 substring(const char str[], int index, int length)的功能是求出字符串 str 中指定序号 index 开始且长度为 length 的子串，并返回所取出的子串。以字符串“China today”为例，其第一个字符“C”的序号为 1(而其在字符数组 str 中的下标为 0)，从序号 5 开始且长度为 3 的子串为“at”。

**【C 函数 1】**

行号	代码
1	char *substring(const char str[], int index, int length)
2	{
3	char *tptr = 0;
4	int k;
5	/*参数错误，返回空指针*/
6	if (index<1    length<0    index+length-1>strlen(str)) return 0;
7	tptr = (char *)malloc((length+1)*sizeof(char));
8	/*申请子串所需的存储区域*/
9	if ( tptr = 0 ) return 0; /*内存申请失败，返回空指针*/
10	for(k=0; k<length; k++)
11	tptr[k] = str[index+k-1];
12	tptr[length-1] = '\0'; /*字符串置结束标志*/
13	return tptr;
14	}

**【问题 1】(6 分)**

函数 substring 中有两处错误，请指出这些错误所在代码的行号，并在不增加和删除代码行的情况下进行修改，写出修改正确后的完整代码行（有注释时，注释可省略）。

出错行号	修改正确后的完整代码行

**【说明 2】**

在 C 函数 2 中, reverse(unsigned intn) 的功能是求出并返回 n 的反序数。例如, 1234 的反序数是 4321, 其求解过程如下:

- (1)由 1234 除以 10 得到商 123 和余数 4, 0 乘以 10 再加上 4 得到 4;
- (2)由 123 除以 10 得到商 12 和余数 3, 4 乘以 10 再加上 3 得到 43;
- (3)由 12 除以 10 得到商 1 和余数 2, 43 乘以 10 再加上 2 得到 432;
- (4)由 1 除以 10 得到商 0 和余数 1, 432 乘以 10 再加上 1 得到 4321。

**【C 函数 2】**

```
unsigned int reverse(unsigned int n)
{
    unsigned int result=0;
    while( (1) ) {
        result= result *10+ n%10,
        n=(2)_;
    }
    return result,
}
```

**【问题 2】(6 分)**

请根据说明 2, 填充 C 函数 2 中的空缺(1)和(2)。

**【问题 3】(3 分)**

用 567, 1234, 56781234, 62354879643 分别作为实参调用函数 reverse, 对应的返回值分别为 765, 4321, 43218765, 1357400630。请说明以 62354879643 作为实参调用函数 reverse 时返回结果出错的原因。

**• (3)**

阅读以下说明和 C 函数, 回答问题 1 和问题 2, 将解答填入答题纸的对应栏内。

**【说明】**

对于具有 n 个元素的整型数组 a, 需要进行的处理是删除 a 中所有的值为 0 的数组元素, 并将 a 中所有的非 0 元素按照原顺序连续地存储在数组空间的前端。下面分别用函数 CompactArr\_v1 和 CompactArr\_v2 来实现上述处理要求, 函数的返回值为非零元素的个数。函数 CompactArr\_v1(int a[], int n) 的处理思路是: 先申请一个与数组 a 的大小相同的动态数组空间, 然后顺序扫描数组 a 的每一个元素, 将遇到的非 0 元素依次复制到动态数组空间中, 最后再将动态数组中的元素传回数组 a 中。

函数 CompactArr\_v2(int a[], int n) 的处理思路是: 利用下标 i (初值为 0) 顺序扫描数组 a 的每一个元素, 下标 k (初值为 0) 表示数组 a 中连续存储的非 0 元素的下标。扫描时, 每遇到一个数组元素, i 就增 1, 而遇到非 0 元素并将其前移后 k 才增 1。

【C 函数 1】	【C 函数 2】
<pre>int CompactArr_v1(int a[], int n) {     int i, k;     int *temp = (int *)malloc( n* (1) );     if (!temp)         return -1;     for(i=0, k=0; i&lt;n; i++)         if ( a[i]!=0 ) {             (2) = a[i];         }     for(i=0; (3); i++)         a[i] = temp[i];     return k; }</pre>	<pre>int CompactArr_v2(int a[], int n) {     int i, k;     for(i=0, k=0; i&lt;n; i++) {         if ( a[i]!=0 ) {             (4) = a[i];         }     }     return k; }</pre>

**【问题 1】 (12 分)**

请根据说明中函数 CompactArr\_v1 的处理思路填补空缺(1)~(3)，根据 CompactArr\_v2 的处理思路填补空缺(4)。

**【问题 2】 (3 分)**

请说明函数 CompactArr\_v1 存在的缺点。

• (4)

阅读以下说明和 C 函数，填补 C 函数中的空缺(1)~(5)，将解答写在答题纸的对应栏内。

**【说明】**

假设一个算术表达式中可以包含以下三种括号：“(”和“)”、“[”和“]”、“{”和“}”，并且这三种括号可以按照任意的次序嵌套使用。

下面仅考虑表达式中括号的匹配关系，其他问题暂时忽略。例如，表达式“[a. (b. 5)] \*c

【{}】”中的括号是完全匹配的，而表达式“【a-(b-5)]\*c”中的括号不是完全匹配的，因为“(”与“】”不能匹配，而且多了一个“(”，即缺少一个与“(”相匹配的“)”。

函数 ifmatched(char expr[])的功能是用栈来判断表达式中的括号是否匹配，表达式以字符串的形式存储在字符数组 expr 中。若表达式中的括号完全匹配，则该函数的返回值为 Matched，否则返回值为 Mismatched。

该函数的处理思路如下：

- (1) 设置一个初始为空的栈，从左至右扫描表达式。
- (2) 若遇上左括号，则令其入栈；若遇上右括号，则需要与栈顶的左括号进行匹配。
- (3) 若所遇到的右括号能与栈顶的左括号配对，则令栈顶的左括号出栈，然后继续匹配过程；否则返回 Mismatched，结束判断过程。
- (4) 若表达式扫描结束，同时栈变为空，则说明表达式中的括号能完全匹配，返回 Matched。

函数 ifMatched 中用到了两种用户自定义数据类型 BOOL 和 STACK，其中，BOOL 类型的定义如下：

```
typedef enum {Mismatched, Matched} BOOL;
```



STACK (即栈类型) 的定义省略, 栈的基本操作的函数原型说明如下:

void InitStack(STACK \*S): 初始化一个空栈。

void Push(STACK \*S, char e): 将一个字符压栈, 栈中元素数目增 1。

void Pop(STACK \*S): 栈顶元素出栈, 栈中元素数目减 1。

char Top(STACK S): 返回非空栈 S 的栈顶元素值, 栈中元素数目不变。

int IsEmpty(STACK S): 若 S 是空栈, 则返回 1, 否则返回 0。

```

【C 函数】
BOOL ifMatched (char expr[])
{
    char *cptr;          /* cptr 指向表达式中的字符 */
    STACK S;
    char e;

    InitStack(&S);      /* 构造一个空栈 */
    for(cptr = expr; *cptr != '\0'; (1)) {
        if (*cptr == '(' || *cptr == '[' || *cptr == '{')
            (2);
        else
            if (*cptr == ')' || *cptr == ']' || *cptr == '}') {
                if (IsEmpty(S))
                    return Mismatched;
                e = (3); /* 取栈顶的左括号 */
                if (*cptr == ')' && e != '(') return Mismatched;
                if (*cptr == ']' && e != '[') return Mismatched;
                if (*cptr == '}' && e != '{') return Mismatched;
                (4); /* 栈顶的左括号出栈 */
            }
    }
    if ((5)) return Matched;
    return Mismatched;
}
    
```

• (5)

阅读以下说明、图和 C++ 代码, 填补 C++ 代码中的空缺(1)~(5), 将解答写在答题纸的对应栏内。

**【说明】**

已知对某几何图形绘制工具进行类建模的结果如图 5.1 所示, 其中 Shape 为抽象类(应至少包含一个纯虚拟(virtual)函数), 表示通用图形, Box 表示矩形, Ellipse 表示椭圆, Circle 表示圆(即特殊的椭圆), Line 表示线条。

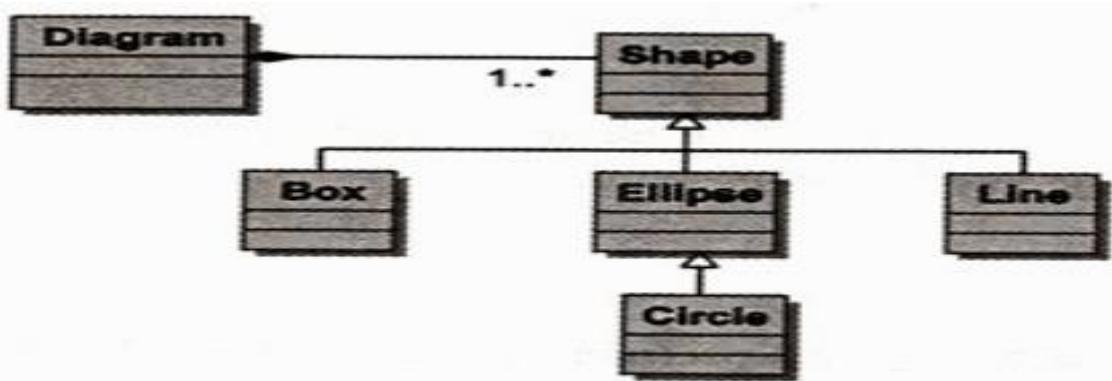


图 6-1 类图

下面的 C++代码用于实现图 5-1 所给出的设计思路, 将其空缺处填充完整并编译运行, 输出结果为:

```

Ellipse
Circle
Ellipse
C
E

```

## 【C++代码】

```

#include <string>
#include <iostream>
using namespace std;
class Shape {
public:
    Shape(const string& name){
        m_name= name;
    }
    ~Shape(){}
    (1) void paintO = 0;
    stringgetNameOconst {
        return name;
    }
private;
    string m_name;
};
//Box 和 Line 类的定义与 Ellipse 类似, 其代码略
class Ellipse (2) {
public:
    Ellipse(const string& name) : Shape(name){ cout<<"Ellipse" <<endl; }
    void paintO { cout<<getNameO<<endl;}
};
class Circle (3) {
public:
    Circle(const string& name) : Ellipse(name){ cout<<"Circ" <<endl; }
};
class Diagram {
public:
    void drawAShap(Shape* shape){ shape->paintO; }
    void drawShapesO {

```



```

    shapes[0] = new Circle("C");
    shapes[1] = new Ellipse("E");
    for (int i=0;i<2; ++1) {
    drawAShap(shapes[i]);
    }
    }
    void close O{ /*删除形状, 代码略 */}
private:
    Shape* shapes[2];
};
int main()
{
    Diagram* diagram = (4)
    diagram->drawShapes0;
    diagram->close O;
    (5) diagram;
}

```

• (6)

阅读以下说明、图和 Java 代码, 填补 Java 代码中的空缺(1)~(6), 将解答写在答题纸的对应栏内。

**【说明】**

已知对某几何图形绘制工具进行类建模的结果如图 6.1 所示, 其中 Shape 为抽象(abstract)类, 表示通用图形, Box (矩形)、Ellipse (椭圆) 和 Line (线条) 继承(extends)了 Shape 类, 其中, Circle 表示圆 (即特殊的椭圆)。

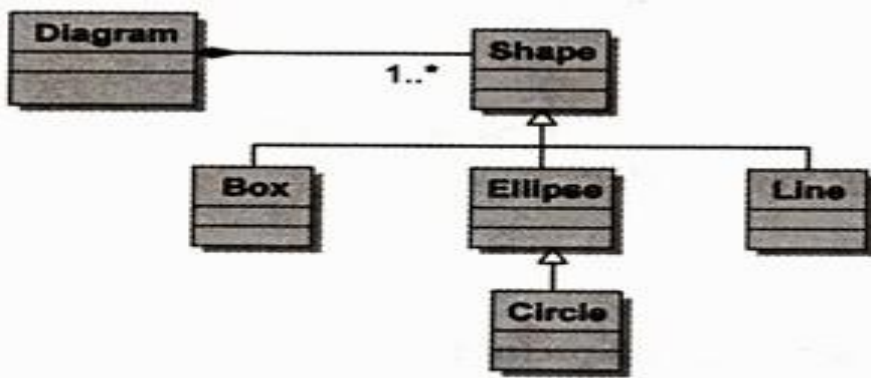


图 6-1 类图

下面的 Java 代码用于实现图 6-1 所给出的设计思路, 将其空缺处填充完整并编译运行, 输出结果为:

```

    Ellipse
    Circle
    Ellipse
    C
    E

```

**【Java 代码】**

```

(1) class Shape{
    public Shape(String name){
    this.name= name;

```

```

    }
    (2) void paint0;
    String getName(){
return this.name;
    }
    final String name;
};
//Box 和 Line 类似下面 Ellipse, 其代码略
class Ellipse (3) {
    public Ellipse(String name){
        super(name);
        System.out.println("Ellipse");
    }
    void paint0{// 绘制现状示意代码
        System.out.println(getName0);
    }
};
class Circle (4) {
public Circle(String name){
    super(name);
    System.out.println("Circle");
}
};
class Diagram {
    private Shape shapes[]= new Shape[2];
    public void draw A Shape(Shape shape){
shape.paint0;
    }
    void erase A Shape(Shape shape){
        // 删除形状, 代码略
    }
    void drawShapes0{
        shapes[0]= new Circle("C");
        shapes[1]= new Ellipse("E");
        for (int i=0; i<2;++i) {
            draw A Shap (shapes[i]); // 绘制形状
        }
    }
    void close0{
        for (int i=0;i<2; ++1) { []关闭图, 删除所绘制图形
            (5) ;
        }
    }
    public static void main(String[] args){
        Diagram diagram= (6) ;
        diagram.drawShapes0;
        diagram.close0;
    }
}

```

希赛网在线题库