

全国计算机技术与软件专业技术资格（水平）考试

2008 年上半年 程序员 下午试卷

（考试时间 14:00~16:30 共 150 分钟）

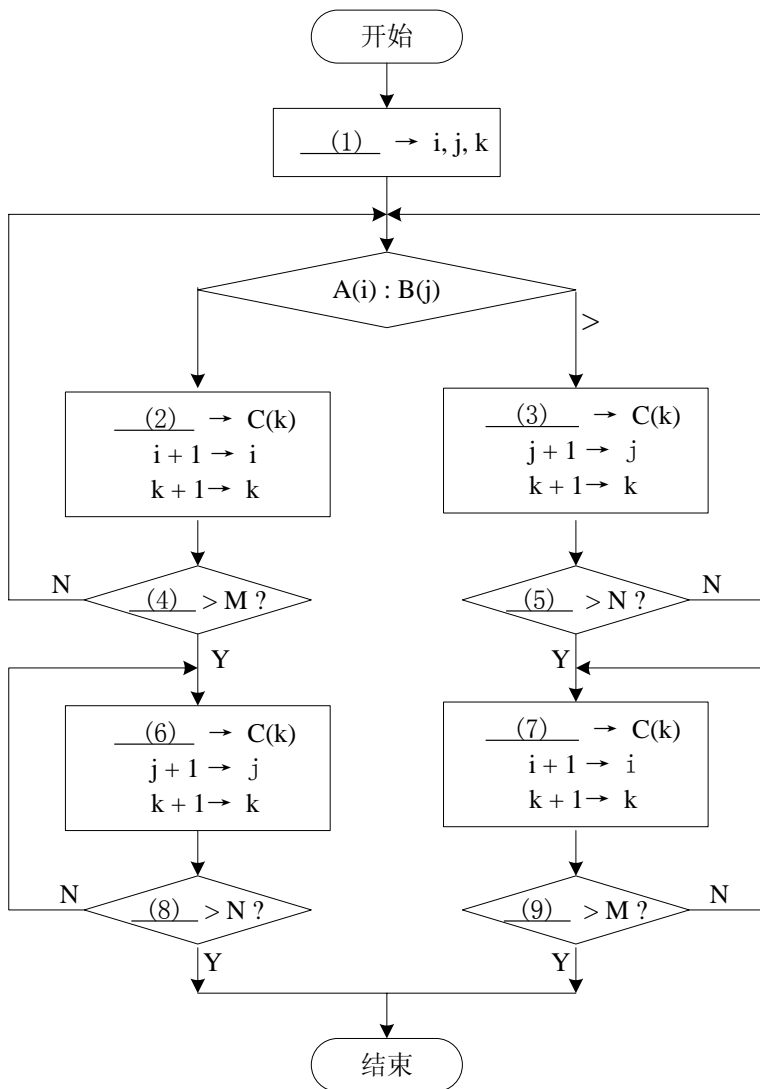
试题一（共 15 分）

阅读以下说明和流程图，填补流程图中的空缺（1）～（9），将解答填入答题纸的对应栏内。

[说明]

假设数组 A 中的各元素 $A(1), A(2), \dots, A(M)$ 已经按从小到大排序 ($M \geq 1$)；数组 B 中的各元素 $B(1), B(2), \dots, B(N)$ 也已经按从小到大排序 ($N \geq 1$)。执行下面的流程图后，可以将数组 A 与数组 B 中所有的元素全都存入数组 C 中，且按从小到大排序（注意：序列中相同的数全部保留并不计排列顺序）。例如，设数组 A 中有元素：2, 5, 6, 7, 9；数组 B 中有元素：2, 3, 4, 7；则数组 C 中将有元素：2, 2, 3, 4, 5, 6, 7, 7, 9。

[流程图]



试题二（共 15 分）

阅读以下说明和C程序，将应填入__(n)__处的字句写在答题纸的对应栏内。

[说明]

下面的程序按照以下规则输出给定名词的复数形式：

- 若名词以“y”结尾，则删除 y 并添加“ies”；
- 若名词以“s”、“ch”或“sh”结尾，则添加“es”；
- 其他所有情况，直接添加“s”。

[C 程序]

```
#include <stdio.h>
#include <string.h>
char *plural(char *word)
{
    int n;
    char *pstr;
    n = strlen(word);          /*求给定单词的长度*/
    pstr = (char *)malloc(n+3); /*申请给定单词的复数形式存储空间*/
    if (!pstr || n < 2)
        return NULL;
    strcpy(pstr, word);      /*复制给定单词*/
    if (__(1)__)
    {
        pstr[n-1] = 'i'; pstr[n] = 'e'; pstr[n+1] = 's'; __(2)__;
    }
    else
        if(pstr[n-1]=='s' || pstr[n-1]=='h' && (__(3)__))
        {
            pstr[n] = 'e'; pstr[n+1] = 's'; pstr[n+2] = '\0';
        }
        else
            { pstr[n] = 's'; pstr[n+1] = '\0'; }
    __(4)__;
}

main( )
{
    int i; char *ps;
    char wc[9][10] =
    {"chair", "dairy", "boss", "circus", "fly", "dog", "church", "clue", "dish"};
    for(i = 0; i < 9; i++) {
        ps = __(5)__;
        printf("%s:  %s\n", wc[i], ps); /*输出单词及其复数形式*/
        free(ps); /*释放空间*/
    }
    system("pause");
}
```

试题三（共 15 分）

阅读以下说明和C程序，将应填入 (n) 处的字句写在答题纸的对应栏内。

[说明]

下面的程序用Dole Rob算法生成N阶（N为奇数）魔方阵（各行、列、对角线数字之和相等）。该算法的过程为：从 1 开始，按如下方法依次插入各自然数，直到 N^2 为止：

- 在第一行的正中插入 1；
- 新位置应当处于最近插入位置的右上方，若该位置已超出方阵的上边界，则新位置取应选列的最下一个位置；若超出右边界，则新位置取应选行的最左一个位置；
- 若最近插入的元素是 N 的整数倍，则选同列的下一行位置为新位置。

例如，3 阶魔方阵如下所示：

8	1	6
3	5	7
4	9	2

[C 程序]

```
#include <stdio.h>
#include <stdlib.h>
#define SIZE 50
main()
{
    int row, col, n, value;
    int a[SIZE+1][SIZE+1]; /*不使用下标为 0 的元素*/
    printf("请输入要输出魔方阵的阶数 n(奇数, <%d):n=", SIZE);
    scanf("%d", &n);
    if (!(n % 2) || n < 1 || (1)) {
        printf("输入数据有误!\n"); exit(0);
    }
    row = 1; col = (n+1)/2; value = 1;
    while(value <= (2)) {
        a[row][col] = value;
        /*计算下一位置*/
        if(value%n != 0) {
            row--; (3);
            if(row < 1) row = n;
            if(col > n) (4);
        }
        else row++;
        value = (5);
    }
    printf("\n%d 阶魔方阵如下所示:\n\n", n);
    for(row = 1; row <= n; row++) {
        for(col = 1; col <= n; col++)
            printf("%5d", a[row][col]);
        printf("\n");
    }
}
```

试题四（共 15 分）

阅读以下说明和C函数，将应填入 (n) 处的字句写在答题纸的对应栏内。

[说明]

计算机在处理算术表达式时，首先将其转换为后缀表达式。例如，表达式“46+5*(120-37)”的后缀表达式形式为“46 5 120 37 - * +”。

计算后缀表达式时，从左至右扫描后缀表达式：若遇到运算对象，则压入栈中；遇到运算符，则从栈中弹出相关运算对象进行计算，并将运算结果压入栈中，重复以上过程，直到后缀表达式扫描结束。例如，后缀表达式“46 5 120 37 - * +”的计算过程为：

- a. 依次将 46、5、120、37 压入栈中；
- b. 遇到“-”，取出 37、120，计算 120 - 37，得 83，将其压入栈中；
- c. 遇到“*”，取出 83、5，计算 5*83，得 415，将其压入栈中；
- d. 遇到“+”，取出 415、46，计算 46+415，得 461，将其压入栈中；
- e. 表达式结束，则计算过程完成。

函数 computing(char expr[], int *result) 的功能是基于栈计算后缀形式的表达式（以串形式存入字符数组 expr）的值，并通过参数 result 返回该值。函数的返回值为 -1/0 分别表示表达式有/无错误。假设表达式中仅包含数字、空格和算术运算符，其中所有项均以空格分隔，且运算符仅包含加（“+”）、减（“-”）、乘（“*”）、除（“\”）。

函数 computing 中所用栈的基本操作的函数原型说明如下：

void InitStack(STACK *s)：初始化栈。

void Push(STACK *s, int e)：将一个整数压栈，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。

int Top(STACK s)：返回非空栈的栈顶元素值，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]

```
int computing(char expr[], int *result)
{
    STACK s;   int tnum, a,b;   char *ptr;
    InitStack(&s);
    ptr = expr;                               /*字符指针指向后缀表达式串的第一个字符*/
    while (*ptr!='\0') {
        if (*ptr==' ') {                       /*当前字符是空格*/
                (1) ;                          /*字符指针指向下一字符*/
            continue;
        }
        else
```

```

if (isdigit(*ptr)) {
    /*当前字符是数字, 则将该数字开始的数字串转换为数值*/
    tnum = (2);
    while (*ptr>=' 0' && *ptr <=' 9' ) {
        tnum = tnum * 10 + (3);
        ptr++;
    }
    Push((4));
}
else /*当前字符是运算符或其他符号*/
    if (*ptr=='+' || *ptr=='-' || *ptr == '*' || *ptr == '/') {
        if (!IsEmpty(s)) {
            a = Top(s); Pop(&s); /*取运算符的第二个运算数*/
            if (!IsEmpty(s)) {
                b = Top(s); Pop(&s); /*取运算符的第一个运算数*/
            }
            else return -1;
        }
        else return -1;
        switch (*ptr) {
            case '+' : Push(&s, b+a); break;
            case '-' : Push(&s, b-a); break;
            case '*' : Push(&s, b*a); break;
            case '/' : Push(&s, b/a); break;
        }
    }
    else
        return -1;
ptr++; /*字符指针指向下一字符*/
} /* while */
if (IsEmpty(s)) return -1;
else {
    (5) = Top(s); Pop(&s); /*取运算结果*/
    if (!IsEmpty(s)) return -1;
    return 0;
}
}

```

从下列 3 道试题（试题五至试题七）中任选 1 道解答。如果解答的试题数超过 1 道，则题号小的 1 道解答有效。

试题五（共 15 分）

阅读下列说明、图和C++代码，将应填入 (n) 处的字句写在答题纸的对应栏内。

[说明]

已知对某载客车辆(Car)进行类建模，如图 5-1 所示，其中类 Engine 表示发动机引擎，类 Wheel 表示车轮，类 Body 表示车身，类 Driver 表示司机，类 Passenger 表示乘客。

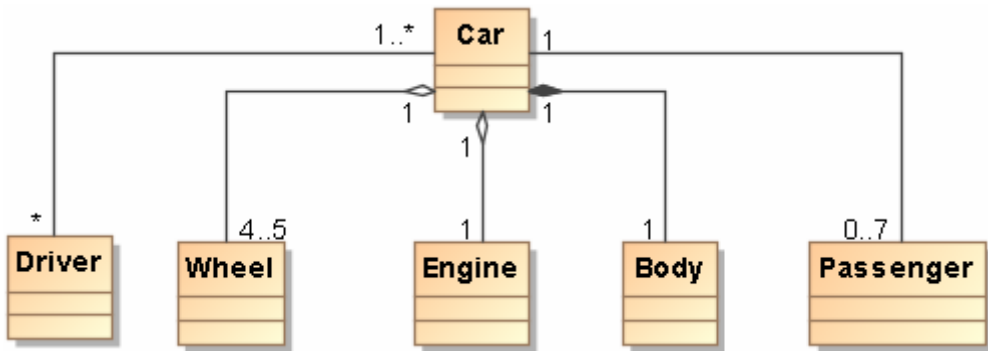


图 5-1 类图

[C++代码]

```

const int (1) = 7;           //定义最多载客数
const int MAX_WHEELS = 5;     //定义最多轮胎数

class Body{                   //此处代码省略 };           //车身类
class Passenger{             //此处代码省略 };           //乘客类
class Wheel{                 //此处代码省略 };           //车轮类

class Driver{ //司机类
public:
    string name; //表示第几路公交车司机
    Driver(string driverName):name((2)){}; //构造函数
};

class Engine{ //引擎类
public:
    string engineNo; //引擎编号
    Engine(string engineNo){ (3) ->engineNo = engineNo; } //构造函数
};
  
```

```

class Car{ //汽车类
protected:
    Engine * engine;    Driver * driver;    Body body;
    Wheel * wheels[MAX_WHEELS];    Passenger * passengers[MAX_PASSENGERS];
public:
    Car(Driver *driver){ //构造函数
        this->driver = driver;
        engine = new Engine("TX6536 型号引擎");
        for (int index = 0; index < MAX_WHEELS; index++){
            wheels[index] = new Wheel();
        }
        for (int index = 0; index < MAX_PASSENGERS; index++){
            passengers[index] = NULL;
        }
    }
    virtual ~Car(){ //析构函数
        for (int index=0; index < MAX_WHEELS; index++){
            delete wheels[index];
        }
        delete (4);
    }
    int getPassengerNumber(){ //获取车上乘客数量
        //此处代码省略
    }
    void getOnPassenger(Passenger * aPassenger ){ //乘客上车
        //此处代码省略
    }
    void run(){ //开车
        if(driver == NULL){ cout << "司机尚未上车!";    return; }
        //此处代码省略
    }
};

void main(){
    Driver driver("第五路公交车司机");
    Car car((5));
    Passenger passengers[MAX_PASSENGERS];
    for (int index = 0 ; index < MAX_PASSENGERS; index ++ ) //乘客上车处理
        car.getOnPassenger (&passengers[index]);
    car.run();
}

```

试题六（共 15 分）

阅读以下应用说明以及 Visual Basic 程序代码，将应填入 (n) 处的字句写在答题纸的对应栏内。

[应用说明]

某应用程序可选择打开用户指定的文本文件，将其内容显示在指定的文本框内供用户编辑，并将编辑后的结果保存在用户指定的文件中。运行时的窗口如图 6-1 所示，其中有六个标签、一个驱动器列表框、一个目录列表框、一个文件列表框、一个文件类型组合框、一个文件编辑文本框、一个文件名文本框以及两个命令按钮。

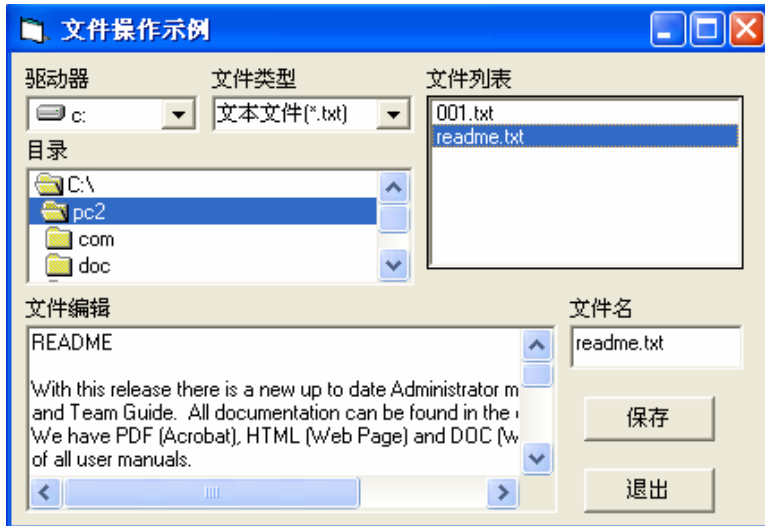


图 6-1

该程序的开发要求如下：

- (1) 通过驱动器列表框 (Drive1)、目录列表框 (Dir1) 和文件列表框 (File1)，选择文件。
- (2) 文件类型组合框 (Cmb_type) 设置为下拉式列表框，其中有三个供选项，分别为“所有文件 (*.*)”、“文本文件 (*.txt)”和“可执行文件 (*.exe)”。在文件列表框中列出的文件类型会自动与文件类型组合框中选择的文件类型相匹配。
- (3) 在文件列表框中单击一个文件名时，该文件名会显示在文件名文本框 (Txt_filename) 中。
- (4) 在文件列表框中双击一个文件名时，若是文本文件，则在文件编辑文本框 (Txt_file) 中显示该文件的内容并可进行编辑；若不是文本文件，则弹出一个对话框，提示“请选择文本文件！”
- (5) 对于编辑后的文本文件，可在文件名文本框 (Txt_filename) 中输入新的文件名，并单击命令按钮 (Cmd_save) 进行保存。

[Visual Basic 程序代码]

```
Private Sub Form_Load()  
    Cmb_type.AddItem "所有文件 (*.*)"   
    Cmb_type.AddItem "文本文件 (*.txt)"
```

```

    Cmb_type.AddItem "可执行文件(*.exe) "
    Cmb_type.ListIndex = 0
    File1.Pattern = "*. *": Txt_filename.Text = ""
    Txt_file.Text = ""
End Sub
Private Sub Dir1_Change()
    File1.Path = (1)
End Sub
Private Sub Drive1_Change()
    Dir1.Path = Drive1.Drive
End Sub

Private Sub Cmb_type_click()
    Select Case Cmb_type. (2)
        Case 0
            File1.Pattern = "*. *"
        Case 1
            File1.Pattern = "*.txt"
        Case 2
            File1.Pattern = "*.exe"
    End Select
End Sub
Private Sub Cmd_save_Click()
    usrFile = GetFileName() '函数 GetFileName 获得要保存的文件名
    Open usrFile For Output As #1 '定义 usrFile 为 1 号输出文件
    Print #1, Txt_file.Text '输出到 1 号文件
    Close #1
End Sub
Private Sub File1_DblClick()
    If right(File1.FileName, 3) <> (3) Then
        MsgBox "请选择文本文件!"
        Exit Sub
    End If
    usrFile = GetFileName() '函数 GetFileName 获得要打开的文件名
    Open usrFile For Input As #1 '定义 usrFile 为 1 号输入文件
    Txt_file.Text = ""
    Do While (4) EOF(1)
        Line Input #1, fContext '从 1 号文件读入一行
        Txt_file.Text = Txt_file.Text + (5) + vbCrLf
    Loop
    Close #1
End Sub
'其他代码略

```

试题七 (共 15 分)

阅读下列说明、图和Java代码，将应填入 (n) 处的字句写在答题纸的对应栏内。

[说明]

已知对某载客车辆(Car)进行类建模，如图 7-1 所示，其中类 Engine 表示发动机引擎，类 Wheel 表示车轮，类 Body 表示车身，类 Driver 表示司机，类 Passenger 表示乘客。

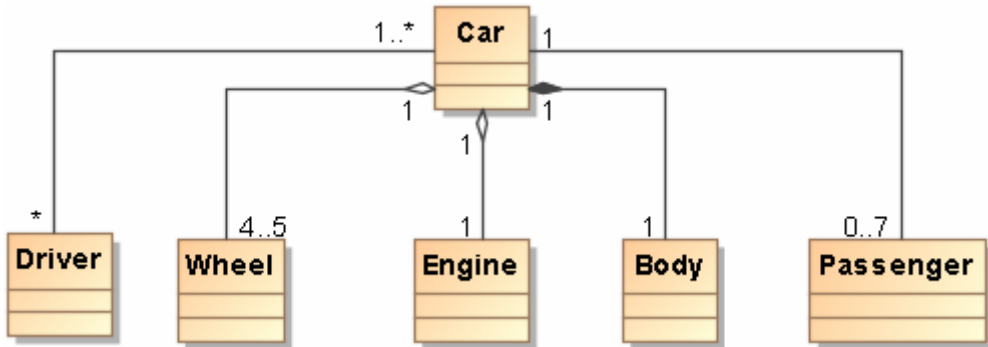


图 7-1 类图

[Java 代码]

```
class Body{          //此处代码省略    }; //车身类
class Passenger{    //此处代码省略    }; //乘客类
class Wheel{       //此处代码省略    }; //车轮类

class Driver{ //司机类
    public String name; //表示第几路公交车司机
    public Driver(String driverName){name = driverName;} //构造函数
};

class Engine{ //引擎类
    public String engineNo; //引擎编号
    public Engine(String engineNo){ this.engineNo = engineNo; } //构造函数
};

public class Car{ //汽车类
    static final int     (1) = 7; //定义最多载客数
    static final int MAX_WHEELS = 5; //定义最多轮胎数
    protected Engine engine;
    protected Driver driver;
    protected Body body = new Body();
    protected Wheel[] wheels;
```

```

protected Passenger[] passengers;
public Car(Driver driver){ //构造函数
    (2) .driver = driver;
    engine = new Engine("TX6536 型号引擎");
    wheels = new Wheel[MAX_WHEELS];
    passengers = new Passenger[MAX_PASSENGERS];
    for (int index = 0; index < MAX_WHEELS; index++){
        wheels[index] = new Wheel();
    }
    for (int index = 0; index < MAX_PASSENGERS; index++){
        passengers[index] = null;
    }
}

int getPassengerNumber(){ //获取车上乘客数量
    //此处代码省略
}

void getOnPassenger(Passenger aPassenger ){ //乘客上车
    //此处代码省略
}

void run(){ //开车
    if( (3) ){ System.out.println("司机尚未上车!"); return;}
    //此处代码省略
}

public static void main(String args[]){
    Driver driver = new Driver("第五路公交车司机");
    Car car = new Car( (4) );
    for (int index = 0 ; index < MAX_PASSENGERS; index ++){
        car.getOnPassenger( (5) Passenger());
    }
}
}

```