

机密★启用前

西南交通大学 2022 年硕士研究生
招生入学考试试卷

试题代码: 840
试题名称: 数据结构与程序设计

考试时间: 2021 年 12 月

考生请注意:

1. 本试题共四题, 共 12 页, 满分 150 分, 请认真检查;
2. 答题时, 请直接将答题内容写在考场提供的答题纸上, 答在试卷上的内容无效;
3. 请在答题纸上按要求填写试题代码和试题名称;
4. 试卷不得拆开, 否则遗失后果自负。

一、单项选择题 (共 52 分, 共 26 小题) (答在试卷上的内容无效)

- ◆ 算法分析的目的是 (1) ____。
A. 找出数据结构的合理性 B. 分析算法的效率以求改进
C. 研究输入和输出的关系 D. 分析算法的可读性
- ◆ 在一个长度为 n 的顺序表中删除第 i 个元素的平均时间复杂度为 (2) ____。
A. $O(1)$ B. $O(\log n)$ C. $O(n)$ D. $O(n^2)$
- ◆ 设堆栈的初始状态为空, 入栈序列为 1, 2, 3, 4, 5, 6, 若出栈序列为 2, 4, 3, 6, 5, 1, 则操作过程中栈中元素个数最多时为 (3) ____。
A. 2 个 B. 3 个 C. 4 个 D. 6 个
- ◆ 设指针 q 指向单链表中结点 X , 指针 p 指向单链表中结点 X 的后继结点 Y , 指针 s 指向被插入的结点 S , 则在结点 X 和结点 Y 之间插入结点 S 的操作序列为 (4) ____。
A. $s->next=p->next; p->next=s;$ B. $q->next=s; s->next=p;$
C. $p->next=s->next; s->next=p;$ D. $p->next=s; s->next=q;$
- ◆ 采用邻接矩阵存储一个无向图, 则该邻接矩阵是一个 (5) ____。
A. 上三角矩阵 B. 稀疏矩阵 C. 带状矩阵 D. 对称矩阵
- ◆ 下列 C 语言程序段的时间复杂度为 (6) ____。

```
int x=1, y=0, a=0, b=0;
```

共 12 页, 第 1 页

```
void f(int n){
    int m=1;
    while(m<=n) m=3*m;
}
```

- A. $O(\log n)$ B. $O(n \log n)$ C. $O(n)$ D. $O(n^3)$
- ◆ 构造散列表的目的是 (7) ____。
A. 插入 B. 删除 C. 快速查找 D. 排序
- ◆ 具有 n 个顶点、 e 条边的无向图的邻接矩阵中, 零元素的个数为 (8) ____。
A. e B. $2e$ C. n^2-1 D. n^2-2e
- ◆ 哈夫曼 (Huffman) 二叉树叶子结点数为 m , 则该哈夫曼树中 NULL 指针域的数目为 (9) ____。
A. $2m$ B. $2m-1$ C. $2m+1$ D. $4m$
- ◆ 从空树开始, 依次插入元素 52, 26, 14, 32, 71, 60, 93, 58, 24, 41 后构成了一颗二叉排序树, 在该树查找 60 要进行比较的次数为 (10) ____。
A. 3 B. 4 C. 5 D. 6
- ◆ 以下平均时间复杂度为 $O(n \log n)$ 且稳定的排序算法是 (11) ____。
A. 快速排序 B. 堆排序 C. 希尔排序 D. 2 路归并排序
- ◆ 以下 4 种内部排序算法中, 元素的移动次数与关键字的初始排列次序无关的是 (12) ____。
A. 直接插入排序 B. 冒泡排序 C. 基数排序 D. 快速排序
- ◆ 假定长度为 k 的关键字序列中全部 k 个关键字均为同义词, 若用线性探测法把该序列 k 个关键字存入该散列表中, 则总共要进行 (13) ____ 次关键字比较 (注意: 判断元素位置空闲时, 才插入元素; 空闲位置的判断要计算为一次比较)。
A. $k-1$ B. k C. $k+1$ D. $k(k+1)/2$
- ◆ 合法且有效的 C 语言常量是 (14) ____。
A. 3E3.0 B. '119' C. '\n' D. 0x119
- ◆ C 语言语句: `int x=20, y='F'-A'; x/=y+1;` 执行代码后, x 的值 (15) ____。
A. 0 B. 20 C. 3 D. 5
- ◆ C 语言中有 `return`, `exit(0)`, `break`, `continue` 可以中止某种代码执行, 只终止函数执行不终止程序执行的关键词是 (16) ____。
A. `return` B. `exit(0)` C. `break` D. `continue`
- ◆ 执行完下面代码之后, a , b 的值是 (17) ____。

```
int x=1, y=0, a=0, b=0;
```

共 12 页, 第 2 页

- ```

while(a<3)
{
 switch(x)
 {
 case 1: switch(y)
 {
 case 0: a++;break;
 case 1: b++;
 }
 case 2: a++;b++;break;
 case 3: a++;b++;
 }
}
printf("a=%d,b=%d\n",a,b);
A. a=3,b=2 B. a=4,b=2
C. a=3,b=0 D. a=4,b=4

```
- 下面代码执行的结果是 (18)。
- ```

int fun(int i)
{
    static int sum=0;
    sum=sum+i; return sum;
}
int main()
{
    int i=1,sum=0;
    while(i<5) sum=fun(i), i++;
    printf("i=%d,sum=%d",i,sum);
}
A. i=5,sum=10        B. i=5,sum=5
C. i=2,sum=0          D. 代码一直运行, 没有输出

```
- 二维数组的元素的初始化错误的是 (19)。
- A. int a[3][3]={ (1, 2, 3), {4, 5, 6} };
- B. int d[2][3]={ (1, 2, 3), {4, 5, 6} };
- C. int c[4][3]={ (1, 2), {4, 5, 6}, {7} };
- D. int b[3][3]={ (1, 2), {4} };
- 定义三维数组 int a[4][5][6]; a[0][0][0]为第 1 个元素, 则第 70 个元素的下标是 (20)。
- A. a[2][2][4] B. a[2][1][4]
- C. a[2][1][3] D. a[2][2][3]
- 函数 void fun1(int a[][4],int n), 定义 int a[3][4], 正确调用 fun1 函数语句是 (21)。
- A. fun1(a+0,3) B. fun1(a[0],3)

- C. fun(*a,3) D. fun(a[0][0],3)
- 下面程序输出结果是 (22)。
- ```

void fun1(int *x) { x=x+1; }
void fun2(int *x) { *x=*x+1; }
int main()
{
 int a[5]={3,1},*p=a+1;
 fun1(p); fun2(p);
 printf("%d",*p);
}
A. 1 B. 2 C. 3 D. 4

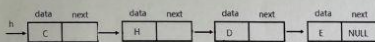
```
- 结构体定义语句, 表述正确的说法是 (23)。
- ```

typedef struct StuInfo
{
    int code;
    char name[20];
}STUDENT;

```
- A. STUDENT 是结构体变量名
- B. 上述定义中不能省略 StuInfo
- C. STUDENT 等价于 struct StuInfo
- D. typedef 不支持如上语句, 存在语法错误。
- 结构体定义如下, 结构体成员 code 引用不正确的代码是 (24)。
- ```

struct StuInfo
{
 int code;
 char name[20];
}stu[20];*p=stu;
A. (*p).code B. stu->code C. p->code D. stu.code

```
- 以下对单向链表结构表述方式正确的是 (25)。
- A. 单向链表结构实质是一种线性表的顺序存储方式。
- B. 单向链表结构是一种动态结构, 可以随机存取节点内容。
- C. 单向链表结构是线性的链式存储结构, 前后节点有逻辑关系。
- D. 单向链表结构的存储单元是连续的, 可以方便增删节点。
- 下图是不带头结点的单向链表结构, 指针变量 h、q、p 分别指向图示结点, 指针变量 h 是头指针指向链表第一个结点, 执行 char temp; temp=p->data; p->data=q->data; q->data=temp; p->next=q->next; 按节点顺序数据域内容是 (26)。



A. CHDE B. CHED C. CHD D. CHE

## 二、填空题 (共 30 分, 共 30 空) (答在试卷上的内容无效)

- 数据元素之间的关系在计算机中有两种不同的表示方法: 顺序映像和非顺序映像, 并由此得到两种不同的存储结构: 顺序存储结构和 (1) 存储结构。
- 查找记录时, 若关键字可以唯一地标识一个记录, 则称此关键字为 (2) 关键字, 否则称为次要关键字。
- 非循环的单向链表结点指针域名为 next, 已知某结点的地址为 p, 判断 p 所指向的结点不为 NULL 且为单向链表最后一个结点的逻辑表达式是 (3)。
- 广义表  $A = (a, (0, b, c), (d, e), f)$ , 则  $\text{head}(\text{tail}(A)) =$  (4)。
- 当用长度为  $n$  的一维数组  $s[0..n-1]$  存储一个堆栈, 若  $\text{top} = n$  表示栈空, 元素  $e$  成功入栈的操作为  $s[\text{--top}] = e$ , 则表示栈满的逻辑表达式是 (5), 栈顶元素成功出栈 (退栈元素存入变量  $e$ ) 的操作是 (6)。
- $n$  个结点  $e$  条有向边的有向图十字链表存储结构中, 表 (边) 结点有 (7) 个。
- 某高级语言 15 行  $\times$  10 列二维数组按“列序为主序”存储各元素, 每个元素占 1 个存储单元, 二维数组行、列下标均从 0 开始, 已知行、列下标分别为 2 (行)、3 (列) 的元素存储地址为 100, 则行、列下标分别为 10 (行)、8 (列) 的元素的存储地址为 (8)。
- 约瑟夫问题:  $m$  个人依次编号为  $1 \sim m$ , 他们围成一圈, 从编号为 1 的人开始沿编号增加方向报数, 数到  $n$  的人出圈, 然后继续从 1 开始报数, 数到  $n$  的人继续出圈, 这样直到圈中只剩下最后一人, 以下算法函数, 用单向循环链表求解该问题, 函数返回值为圈中最后一人的编号, 请填写使程序完整。

已知链表结点及结点指针数据类型定义如下:

```
typedef struct node { int no; /* 人的编号 */
 struct node *next;
} LNode, *LPtr;

int Josephu(int m, int n) /* 分析时以 m=8, n=3 为例 */
{ /* 创建单向循环链表 */
 LPtr h, last, p, pr; int i;
 /* 创建单向循环链表 */
```

```
h=(LPtr)malloc(sizeof(LNode)); h->no=1;
last=h;
for(i=2; i<=m; i++)
{ p=(LPtr)malloc(sizeof(LNode)); p->no=(9);
 last->next=(10); last=p;
}
last->next=h;
/* 数到 n 的人出圈(从循环链表中删除), 直到循环链表只剩下一个结点 */
pr=last;
while(pr->next!=pr)
{ for(i=1; i<=(11)-i++; pr=pr->next; /* pr 推进到出圈人的直接前趋 */
 p=pr->next; /* p 指向出圈人 (被删除结点) */
 (12);
 free((void *)p); /* 删除出圈人 */
}
return pr->no; /* 返回圈中最后一人的编号 */
}

* 完全二叉树按结点编号次序采用顺序存储结构实现存储, 已知数组下标从 0
开始 (即根结点的下标为 0), 以下算法利用堆栈消除递归实现完全二叉树的
先序遍历, 请填写使程序完整。
void pre_travel(ElemTp a[], int n) /* n 为完全二叉树结点总数 */
{ int i=0;
 initStack(S); /* 初始化堆栈并置空栈 */
 while((13) || !empty(S)) /* empty(S) 测试栈空 */
 { if(i<n) { visit(a[i]); /* 访问当前子树根结点 a[i] */
 push(S, (14));
 i=2*i+1; /* i 向左分支深入, 指向左儿子 */
 }
 else { i=pop(S);
 i=(15); /* i 向右分支深入, 指向右儿子 */
 }
}

* 整型数值以 (16) 形式存储在计算机内存中。
* 韩信点兵问题: 从 1 至 5 报数, 最后一个士兵报 1; 按从 1 至 6 报数, 最后一
```

个士兵报 5; 从 1 至 7 报数, 最后一个士兵报 4; 从 1 至 11 报数, 最后一个士兵报 10。设士兵数  $x$ , 满足条件为真的表达式: (17)。

- 程序填空: 计算 1~100 所有能被 3 整除数之和。

```
int i=100;
(18);
do
{
 if((19))
 sum=sum+i;
 (20);
}while(i>0);
```

- union Test; int a; float b; double c; } x; 变量 x 在内存中所占的字节数是 (21)。

- 定义一个 3 行 4 列的整型二维数组 a 并初始化, 使得初始化后二维数组的所有元素为 0, 正确的定义语句是 (22)。

- int a[4][3], (\*p)[3]=a; 利用指针 p 为 a[1][2] 赋值为 12 语句为 (23)。

- 完成 int strcmp\_1(char \*str1, char \*str2) 函数: 用于比较两字符串大小, 比较规则是, 两字符串从左至右逐个字符一次比较 (按照 ASCII 码值大小), 直到出现不同字符或者字符串结束标志, 返回两字符串对应字符的差值。 (例如: "abcd" 和 "abce", 返回值是 -3)。

```
int strcmp_1(char *str1, char *str2)
{
 int value=0;
 while(1)
 {
 value= (24);
 if(!value&&*str2)
 (25);
 else
 (26);
 }
 return value;
}
```

- 不带头结点的单向链表操作: 包含一个节点数据的单向链表结构定义。  
typedef struct Node

```
{ int data;
 struct Node *next;
}NODE;
```

指针 p\_new 指向新建节点: NODE \*p\_new= (27);  
显示链表中所有节点的节点号和该节点的中数据项内容。

```
void printAllNode(NODE *head)
{
 NODE *p= (28);
 int i=1;
 while((29))
 {
 printf("%5d%5d\n", i, p->data);
 (30);
 i++;
 }
}
```

三、简答题 (共 48 分, 共 10 小题, 前 4 小题每题 6 分, 后 6 小题每题 4 分) (答在试卷上的内容无效)

- 10 个记录按关键字非递减次序存储于一维数组 a[0..9], 考虑二分 (折半) 查找, 求解以下问题。 (共 6 分)

- (1) 画出对应的二叉判定树;
- (2) 若各记录等概率查找, 计算成功查找时的平均查找长度 (ASL<sub>su</sub>);
- (3) 写出失败查找时, 最大的关键字比较次数。

2. 某整型数组 A 的 10 个元素依次为: 6, 2, 9, 7, 3, 8, 4, 5, 0, 1。用下列各排序方法, 将 A 中元素由小到大排序。 (共 6 分)

- (1) 取第一个元素值 6 作为支点 (枢轴), 试写出快速排序第一次划分后 A 中的结果。
- (2) 用堆排序, 试写出将第一个选出的数据放在 A 的最后位置上, 且将 A 前面 9 个元素重新调整成堆后的 A 中结果。
- (3) 进行增量为 3 的一趟希尔排序, 试写出 A 中结果。

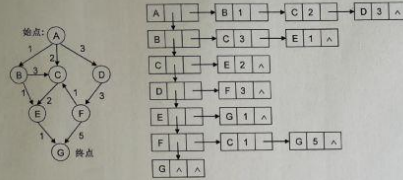
3. 设某段通信电文由 A, B, C, D, E, F 六个字符组成, 它们在电文中出现的次数分别是 16, 5, 9, 3, 20, 1。 (共 6 分)

- (1) 试画出编码用的哈夫曼 (Huffman) 二叉树;
- (2) 对该段电文进行哈夫曼编码, 试计算该段电文的总比特数。



4. 某赋权有向图及它的邻接表如下, 求解以下问题。(共 6 分)

- (1) 基于该邻接表存储结构, 写出从始点 A 出发顶点的深度优先搜索顺序, 画出对应的深度优先生成树;
- (2) 将该图作为 AOE 网络图, 试写出各顶点的最早/最晚开始时间, 并说明工程的最短完成时间。



5. 以下代码段运行时, 从键盘输入 abcdef (箭头表示回车), 则运行的结果是: (4 分)

```
int main()
{
 char ch;
 while((ch=getchar())!='\n')
 {
 if(ch%2)
 ch=ch+2;
 else
 ch=ch+1;
 putchar(ch);
 }
}
```

6. 以下代码段运行的结果是: (4 分)

```
void fun(int b[],int n)
{
 int *p1,*p2,*p;
 p1=b; p2=b+n-1;
```

```
for(int i=0;i<=4;i++)
{
 p=p1; p1=p1+1; p2=p; }
}
int main()
{
 int a[5]={1,3,5,6};
 fun(a,5);
 for(int i=0;i<=4;i++)
 printf("%d",a[i]); // %d 后面有一逗号
}
```

7. 以下代码段运行的结果是: (4 分)

```
int sum=0;
void fun()
{
 static int x=1;
 int i=1;
 do {
 sum=sum+x;
 i=i+1;
 x=x+1;
 while(i<=x/2);
 }
 int main()
 {
 for(int i=1;i<=3;i++)
 fun();
 printf("%d",sum);
 }
```

8. 以下代码段运行的结果是: (4 分)

```
void fun(char *str)
{
 if(*str==0) return;
 else
 {
 fun(str+1);
 if(*str>='9' && *(str)<='122')
 printf("%c",*(str)); } }
```

```
int main()
{
 fun("Thank You");
 return 0;
}
```

9. 以下代码段运行的结果是: (4 分)

```
void fun(char *p, char *s1, char *s2)
{
 while(*s1 != 0 && *s2 != 0)
 {
 if(*s1 < *s2) *p++ = *s1++;
 else *p++ = *s2++;
 while(*s1 != 0) *p++ = *s1++;
 while(*s2 != 0) *p++ = *s2++;
 *p = 0;
 }
 void main()
 {
 char *p1="BDB", *p2="ACA";
 char str[80];
 fun(str,p1,p2);
 puts(str);
 }
```

10. 以下代码段运行的结果是: (4 分)

```
void fun(int a[]) {
 int i=0;
 do { a[i+1]=a[i+1]+a[i];
 i++;
 } while(i<4);
}
int main() {
 int i,a[5]={2,3,4,5,6};
 fun(a);
 for(i=0;i<5;i++)
 printf("%d",a[i]); // %d 后面有一逗号
```

#### 四. 算法与程序设计(共 20 分, 共 2 小题)(答在试卷上的内容无效)

1. 已知二叉树的二叉链表结点及结点指针数据类型定义如下:  
typedef struct node { double data; struct node \*lchild, \*rchild; } BITNode, \*BIT;  
试编写算法函数, 形参传入二叉树根结点指针, 函数返回二叉树中 data 域值小于 0 的结点数目。(10 分)
2. 编写程序, 定义含有 10 个元素一维整型数组, 编写整型数组数据的输入函数, 编写计算数组的最大值、最小值及其所在下标位置的函数, 完成函数定义及 main 函数测试。(10 分)

例如输入数据: 2 5 0 4 1 9 12 45 15 77  
输出结果: 最大值 a[9]=77, 最小值 a[2]=0

要求完成相关函数定义:

- (1) 整型数组数据的输入函数 (2 分);
- (2) 计算数组的最大值、最小值及其所在下标位置的函数 (5 分);
- (3) main()测试函数 (3 分)。