《数据结构》参考答案

一、选择题**（每小题2分，共20分）**

1-5 ABABD

6-10 BBBCC

二、填空题**（每小题2分，共20分）**

1、O(n2)， O(nlog2n)

2、p>llink->rlink=p->rlink;p->rlink->llink=p->llink

3. 3

4、 等于

5、ABDECF，DBEAFC，DEBFCA

6、50，51

7、m-1，(R-F+M)%M

8、n+1-i，n-i

9、(19，18，16，20，30，22)

10、A[i][j]=1

三、名词解释题**（每小题5分，共15分）**

1. 强连通图：在有向图G中，如果对于每一对vi,vj∈V,vi≠vj，从vi到vj和从vj到vi；都存在路径，则称G是强连通图。

强连通分量：有向图中的极大强连通子图称做有向图的强连通分量。

2. 深度优先搜索：假设初始状态是图中所有顶点未曾被访问，则深度优先搜索可从图中某个顶点v出发，访问此顶点，然后依次从v的未被访问的邻接点出发深度优先遍历图，直至图中所有和v有路径相通的顶点都被访问到；若此时图中尚有顶点未被访问，则另选图中一个未曾被访问的顶点作起始点，重复上述过程，直至图中所有顶点都被访问到为止。

3. 不稳定排序：假设Ki=Kj(1 ≤i ≤n,1 ≤j ≤ n,i≠j),且在排序前的序列中Ri领先于Rj(即i＜ j)。若在排序后的序列中Rj领先于Ri,则称所用的排序方法是不稳定的。

稳定排序： 假设Ki=Kj(1 ≤i ≤n,1 ≤j ≤ n,i≠j),且在排序前的序列中Ri领先于Rj(即i＜ j)。若在排序后的序列中Ri仍领先于Rj,则称所用的排序方法是稳定的

四、应用题**（每小题6分，共30分）**

**1.** H(1)=1 Mod 11=1 H(13)=13 Mod 11=2 H(12)=12 Mod 11=1（冲突）

H(34)=34 Mod 11=1（冲突） H(38)=38 Mod 11=5 H(33)=33 Mod 11=0

H(27)=27 Mod 11=5（冲突） H(22)=22 Mod 11=0（冲突）

（1）把关键字依次填入下表，并统计每个关键字查找成功时比较次数（8分）

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 地址 | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
| 关键字 | 33 | 1 | 13 | 12 | 34 | 38 | 27 | 22 |  |  |  |
| 比较  次数 | 1 | 1 | 1 | 3 | 4 | 1 | 2 | 8 |  |  |  |

（2）查找成功时ASL： （2分）

ASL成功=（1+1+1+3+4+1+2+8）/8=21/8。

**2.**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 终点 | 最短路径求解过程 | | | | |
| b | 6 (a,b) | 5 (a,c,b) |  |  |  |
| c | 3 (a,c) |  |  |  |  |
| d |  | 6 (a,c,d) | 6 (a,c,d) |  |  |
| e |  | 7 (a,c,e) | 7 (a,c,e) | 7 (a,c,e) |  |
| f |  |  |  | 9 (a,c,d,f) | 9 (a,c,d,f) |
| vj | c | b | d | e | f |
| S | {a,c} | {a,c,b} | {a,c,d} | {a,c,e} | {a,c,d,f} |

3. WPL=（3+4）×4+（6+7+8）×3+（9+10）×2=129

4.



5．堆排序 建大堆：58,47,51,29,18,12,25,10；

第一趟：51,47,25,29,18,12,10,58；

第二趟：47,29,25,10,18,12,51,58

第三趟：29,18,25,10,12,47,51,58；

第四趟：25,18,12,10,29,47,51,58；

第五趟：18,10,12,25,29,47,51,58；

第六趟：12,10,18,25,29,47,51,58；

第七趟：10,12,18,25,29,47,51,58

**五、设计题（每小题3分，共15分）**

1. void delRandX\_Y(LinkList\* L,TypeData x,TypeData y){

LinkList\* p;

LinkList\* pre;

p = L->next;

pre = L;

LinkList\* temp;

if(L == NULL || L->next == NULL)

return ;

while(p!=NULL){

//找到并删除

if(p->data > x && p->data < y){

temp = p;

pre->next = p->next;

p = p->next;

free(temp);

}else{

pre = pre->next;

p = p->next;

}

}

}

2. LinkList DisCreat\_1(LinkList &A){

i=0;

B=(Linklist)malloc(sizeof(LNode));

B->next=NULL;

LNode \*ra=A,\*rb=B;

p=A->next;

A->next=NULL;

while(p!=NULL){

i++; //序号加1

if(i%2==0){

rb->next=p;

rb=p;

}

else{

ra->next=p;

ra=p;

}

p=p->next;

}

ra->next=NULL;

rb->next=NULL;

return B;

}

3. int Max=0;

int conut\_[MaxSize];

void getBiWid(BiTree \*T,int k){ //k变量为当前层数 默认从第一层开始

if(!T)

return;

conut\_[k]++;

if(conut\_[k]>Max)

Max=conut\_[k]; //Max为数组的最大值

getBiWid(T->rchild,k+1);

getBiWid(T->lchild,k+1);

}