《数据结构》参考答案

一、选择题**（每小题2分，共20分）**

1-5 CDABB

6-10 CACCB

二、填空题**（每小题2分，共20分）**

1、 存储（或存储结构）

2、p－＞next－＞next

3、(x,y,z)

4、三元组 十字链表

5、a4,8

6、384

7、abefcdg

8、log2n+1

9、2

10、n2-2e

三、名词解释题**（每小题5分，共15分）**

1. 线索：在二叉树中 , 利用空余的指针指向二叉树某种遍历方式的结点的前驱和后继, 这种指向前驱和后继的指针, 叫线索。

线索二叉树：对二叉树以某种次序进行遍历并加上线索的过程叫做线索化。线索化了的二叉树称为线索二叉树。

2. 图：一个图 G=（V，E）是一个由非空的有限集 V 和一个边集 E所组成的。若E中的每条边都是顶点的有序对（ v , w），就说该图是有向图。若 E 中的每条边是两个不同顶点无序对，就说该图是无向图，其边仍表示成（ v, w ）

3. 最小生成树：设 G=(V, E )是一个连通图，E中每一条边 (u, v) 的权为 C(u, v)，也叫做边长。图 G的一株生成树 (spanning tree)是连接V中所有结点的一株开放树。将生成树中所有边长之总和称为生成树的价(cost)。使这个价最小的生成树称为图 G的最小生成树。

四、应用题**（每小题6分，共30分）**

**1.** H(22)=22\*3 Mod 13=1 H(41)=41\*3 Mod 13=6 H(53)=53\*3 Mod 13=3

H(46)=46\*3 Mod 13=8 H(30)=30\*3 Mod 13=12 H(13)=13\*3 Mod 13=0

H(1)=1\*3 Mod 13=3（冲突） H(67)=67\*3 Mod 13=6（冲突）H(51)=51\*3 Mod 13=10

（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 | 11 | 12 |
| 关键字 | 13 | 22 |  | 53 | 1 |  | 41 | 67 | 46 |  | 51 |  | 30 |
| 比较  次数 | 1 | 1 |  | 1 | 2 |  | 1 | 2 | 1 |  | 1 |  | 1 |

（2）查找成功时ASL： （2分）

ASL成功=（1+1+1+2+1+2+1+1+1）/9=11/9。

2. 答案：(1)求ve和vl (2)关键路径



3.

构造过程：



4.



WPL：2\*4+5\*4+7\*3+8\*3+9\*3+10\*2+11\*2=142

5. 答案：

第一次排序：（076,129），256,（751.937,863,742,694,301,439）

第二次排序：076,129，256,（751.937,863,742,694,301.439）

第三次排序：076,129,256，（438,301,694.742），751，（863,937）

第四次排序：076,129,256,301,438，（694,742），751，（863,937）

第五次排序：076,129,256,301,438,694,742,751,（863,937）

第六次排序：076,129,256,301,438,694,742,751,863,937

**五、设计题（每小题3分，共15分）**

1．int GetElem(LinkList L,int i,Elemtype \*e){

LinkList p；int j；

p=L->next;j=1;

while(p&&j<i){

p=p->next;++j;

}

if(!p||j>i) return ERROR;

\*e=data;

return OK;

}

2．int ListDelete\_sq(Sqlist \*L,int i){

int k;

if(i<1||i>L->length) return ERROR;

for(k=i-1;k<L->length-1;k++)

L->slist[k]= L->slist[k+1] ;

--L->Length;

return OK;

}

3. typedef struct node {int data; struct node \*lchild,\*rchild;} bitree;

void swapbitree(bitree \*bt)

{

bitree \*p;

if(bt==0) return;

swapbitree(bt->lchild); swapbitree(bt->rchild);

p=bt->lchild; bt->lchild=bt->rchild; bt->rchild=p;

}