**习题2 表结构**

2-1 如果长度为n的线性表采用顺序存储结构存储，则在第i (1≤i≤n+1)个位置插入一个新元素的算法的时间复杂度为( **B** )。

(A) O(1)

(B) O(n)

(C) O(nlog2n)

(D) O(n2)

2-2 在一个有127个元素的顺序表中插入一个新元素，要求保持顺序表元素的原有(相对)顺序不变，则平均要移动( **C** )个元素。

(A) 7

(B) 32

(C) 64

(D) 127

2-3 已知An×n为稀疏矩阵。试从时间和空间角度比较，采用二维数组和三元组顺序表两种存储结构计算∑aij的优缺点。

若稀疏矩阵中有m个非0元素

**三元组：用三个一维数组保存。占用3\*m（若不用链表则3\*n）个空间单元。运算∑aij 时间复杂度为O(m)**

**二维数组：用二维数组保存，两个下标分别存坐标。占用n\*n个空间单元，运算∑aij 时间复杂度为O(n^2)**

2-4 综合比较顺序表和链表。

|  |  |  |
| --- | --- | --- |
|  | **顺序表** | **链表** |
| **实现方法** | **可直接使用数组/动态数组** | **需要设计链表结构，至少有数据域和指针域** |
| **储存方法** | **若使用静态数组，储存范围受限**  **若使用动态数组则每次需要申请一片连续内存空间，麻烦** | **增加节点才申请一个节点的内存空间，方便** |
| **访问指定位次数据** | **可直接通过下标取出数据，时间复杂度O(1)** | **需要从头指针开始遍历，时间复杂度O(n)** |
| **在头部插入数据** | **需要将所有的数据向下标+1方向移动，时间复杂度O(n)** | **无须遍历直接调整指针域即可，时间复杂度O(1)** |
| **在第i位插入数据** | **需要将i+1 ~ n位的数据后移，时间复杂度O(n-i)** | **需要遍历i次，时间复杂度O(i)** |
| **在尾部删除数据** | **直接插入，时间复杂度O(1)** | **若有存尾节点则O(1)，无则O(n)** |
| **在头部删除数据** | **需要将所有的数据向下标-1方向移动，时间复杂度O(n)** | **无须遍历直接调整指针域即可，时间复杂度O(1)** |
| **在第i位删除数据** | **需要将1 ~ i位的数据前移，时间复杂度O(i)** | **需要遍历i次，时间复杂度O(i)** |
| **在尾部删除数据** | **直接插入，时间复杂度O(1)** | **若有存尾节点则O(1)，无则O(n)** |

2-5 解释链表的“头指针、头结点和首元素结点”三个概念。

**头指针：链表的开头，链表由头指针确定，必须通过头指针来找到头指针指向的链表**

**头节点：链表不为空时头指针指向的节点**

**首元素：链表中用于存放数据的第一个节点**

2-6 设链表L→a→b→c→d，指针域为\*next。执行下列命令后，(**p→c→d，q→NULL** )。

p=L->next->next;

L->next->next=NULL;

q=L->next->next;

(A) p→b→c→d，q→a

(B) p→b→c→d，q→NULL

(C) p→c→d，q→a

(D) p→c→d，q→a→b

2-7 描述下列算法的主要功能是( **A** )。

① 构造头结点L，取q=L;

② 产生1个结点p;

③ q−>next=p;

④ 输入p−>data的值;

⑤ 取q=p;

⑥ 重复执行②至⑤n次;

⑦ p−>next=NULL;

(A) 通过输入n个数据元素构建链表L

(B) 采用前插法，在链表L中输入n个数据元素

(C) 通过产生n个结点构建链栈L，q为栈顶指针

(D) 在链队列L中输入n个数据元素，q为队尾指针

2-8 设两个循环链表的长度分别为n和m，则将这两个循环链表连接成一个循环链表，最好的时间复杂度为( **D** )。

(A) O(1)

(B) O(n)

(C) O(m)

(D) O(min(n，m))

2-9 设push和pop分别表示进栈和出栈操作，输入序列为xyz，则经过栈操作( **A** )可以输出序列yzx。

(A) push, push, pop, push, pop, pop

(B) push, push, push, pop, pop, pop

(C) push, pop, push, pop, push, pop

(D) push, pop, push, push, pop, pop

2-10 设进栈序列为123，试给出所有可能的出栈序列。

**123 push pop push pop push pop**

**132 push pop push push pop pop**

**213 push push pop pop push pop**

**231 push push pop push pop pop**

**321 push push push pop pop pop**

2-11 如果进栈序列为123456，能否得到出栈序列435612和135426?

答：435612不可以，135426可以

2-12 简述算法的功能(设数据元素类型为int)：

void proc(LinkQueue \*Q)

{

LinkStack S;

InitStack(S);

while(!EmptyQueue(Q) )

{

DeleteQueue(Q, d);

Push(S,d);

}

while(!EmptyStack(S) )

{

Pop(S, d);

InsertQueue(Q, d);

}

}

**答：通过栈S将队列Q倒置**

2-13 描述下列递归算法的功能。

int F(int m, int n)

{

if (n>m) return F(n, m);

else if (n==0) return m;

else return F(n, m%n);

}

辗转相除法，将m,n的最大公约数作为函数返回值返回

2-14 编写递归算法：

|  |  |  |  |
| --- | --- | --- | --- |
|  |  |  | 0，m=0且n≥0 |
|  | g(m, n)= |  |
|  |  | g(m-1, 2n)+n，m>0且n≥0 |

**答：**

**int g(int m,int n){**

**if(m == 0 && n >= 0){**

**return 0;**

**}**

**if(m > 0 && n >= 0){**

**return g(m-1,2\*n)+n;**

**}**

**}**

2-15 将下列递归过程改写为非递归过程。

void test(int &s)

{

int x;

scanf ("%d", &x);

if (x==0) s=0;

else

{

test(s);

s+=x;

}

}

**答：**

**void test(int &s)**

**{**

**int x;**

**while(scanf("%d",&x) && x != 0){**

**s += x;**

**}**

**s = 0;**

**int x;**

**}**

2-16 按照格式要求给出调用F(3,'A','B','C')的运行结果：

void F(int n, char x, char y, char z)

{

if (n==1) printf("1 %c 🡪 %c\n", x, z);

else

{

F(n-1, x, z, y);

printf("%d %c 🡪 %c\n", n, x, z);

F(n-1, y, x, z);

}

}

1 A -> C

2 A -> B

1 C -> B

3 A -> C

1 B -> A

2 B -> C

1 A -> C