2022年电子科技大学820计算机专业课真题

操作系统

一、选择题

1、那个不是操作系统要考虑的（D）
A 方便　B 有效　C 可扩展　D 容错

2、以下哪种不能实现互斥（D）
A 管程　B 信号量　C 管道

3、以下正确的（C）
A 单道批处理不需要
B 单道批处理不需要存储保护技术
C 单道批处理不需要对换技术
D 多道批处理不需要

4、进程和线程的区别

5、下列正确的是
A 系统调用不会引起模式切换
B 系统调用不会引起进程切换
C 响应外部中断不会引起模式切换
D 响应外部中断不会引起进程切换

6、那个算法会饥饿（B）
A 先来先服务 B 短进程优先 C 最短剩余时间优先 D 时间片轮转

7、考了用户调用和外部中断，会不会导致进程转换和模式转换

8、不会饥饿的算法，就是FCFS

二、填空题

1、索引节点有4个直接块1个一级2个二级3个三级，块大小4KB，地址块大小4B，支持的最大文件长度是$\frac{\left(4+2^{10}+2^{21}+3\times 2^{30}\right)\times 4KB}{}$

给了一个逻辑地址求访问几次磁盘。
2、4个进程1类资源，需要（3,2,9,7）已分配1,1,3,2，求现在有多少才安全（10）
3、8192转每分，每个磁道500扇区一个扇区512B，一个500MB的文件要多少秒（15s）（忽略旋转延迟和寻道时间）.
4、位图，扇区512B，一个簇16扇区，磁盘1TB，问位图大小（16MB）.

三、简答题

1、用 C 伪代码描述信号量递增递减操作

```
void wait(semaphore S){
S.value--;
if (S.value <0){
add this process to S.L;
block(S.L);
}
}
void signal(semaphore S){
S.value++:
if (S.value <=0){
remove a process P from S.L;
wakeup(P);
}
}
```

2、

给了两段代码，代码内容一样的

```
x=5
A
1   x=x-1
2   x=x+1
3   if(x!=5)
4   print x
B
1   x=x-1
2   x=x+1
3   if(x!=5)
4   print x
```

第一问，两个进程能不能互斥访问共享变量 x

答：不能。

第二问给一个只输出一个 5 的执行顺序

A1,B1,A2,A3,B2A4B3B4.(其他也是可以的)

3、给了一个请求存储空间的序列（动态分区分配）。

第一问用动态分区最佳分配问能否满足最后一个请求，不能说明原因，能的话画出满足后空闲分区表。

第二问用伙伴系统问题同上。

4:逻辑地址和物理地址都是 32 位，页面大小 4kb,页表宽度 32 位，其中低 12 位为标识位。

第一问 给出逻辑地址结构

20 位（页号）	12 位（页内偏移量）
31 12 11	0

第二问 给出页表表头结构

20 位（页号）	12 位（页内偏移量）
31 12 11	0

第三问 给了一个逻辑地址还给了很多内存块求物理地址

5:多级反馈队列求周转时间，带权周转时间，平均周转时间，平均带权周转时间。

6:PV 题

图书管理，有 m 种不同的书，每种 N_m 本。

要求：

1、有人查询时，为确保查询结果正确，结果返回前，不准其他人借阅和还书，可以查询。写出借阅，归还和借书进程的代码（可同时查阅，但查阅结果返回前不允许其他同学修改图书信息)

2、若数量大于 0，则可借阅，借阅后数量-1，若为 0 结束本次借阅

3、归还后数量+1

4、有同学借阅、归还时不允许其他同学查阅借阅归还

```
semaphore mutex_all = 1, mutex_count_reference = 1, mutex_count_number[m] = { 1 }, mutex_revise = 1;
int count_reference = 0;
int count_number[m] = { N_1,N_2,...,N_m };
Reference(m) {
        P(mutex_count_reference);
        if (count_reference == 0)
                P(mutex_all)
        count_reference++;
        V(mutex_count_reference);
        P(mutex_revise);
        Reference...;
        Printf(count_number[m]);
        V(mutex_revise);
        P(mutex_count_reference);
        count_reference--;
        if (count_reference == 0)
                V(mutex_all)
        V(mutex_count_reference);
}
```

```
Borrow(m) {
        P(mutex_all);
        P(mutex_count_number[m]);
        if (count_number[m] > 0) {
                count_number[m] = count_number[m] - 1;
                V(mutex_count_number[m]);
        }
        else {
                Ending...;
                V(mutex_count_number[m]);
        }
        V(mutex_all);
}
```

```
Repay(m) {
        P(mutex_all);
        P(mutex_count_number[m]);
        count_number[m] = count_number[m] +1;
        V(mutex_count_number[m]);
        V(mutex_all);
}
```

数据结构

一、选择题

1、查找时，求平均长度

2、一段代码的最坏时间复杂度

A　n 的 3 次

B　n 的 4 次

C　n 的二次乘 logn

```
for (i=0;i<nn;i++){
t+=x*x;
for (j=i,i<n;j++)
t+=i*j;
}
```


3、进程调度算法时间开销最小的(A)

A 短进程优先 B 先来先到 C 多级反馈 D 高响应比优先

4 一个基本有序（无序）的序列用那种方法时间开销最小(AC)

A 冒泡　B 希尔　C 快速　D 堆

二、填空题

1、m 颗完全二叉树，每颗高度 hi，最多结点数 $\sum_{i=1}^{m}\left(2^{h_i}-1\right)$，最少结点数 $\sum_{i=1}^{m}\left(2^{h_i-1}\right)$

2、权重 3 5 6 12 89 71 52，3 和 6 编码长度最短是 6、5.

3、1 个度为 1，2 个度为 2，3 个度为 3，4 个度为 4，问叶子结点 22.

三、简答题

1、H(key)=key%17，表长为 20，对 12 15 39 55 14 9 38 构造散列表，用二次探测再散列法处理冲突（即 d=1,－1,4,－4,...），画出构造的散列表。

地址	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19
key				38	55	39				9			12		14	15				
比较次数				3	1	1				1			1		1	1				

2、两个字符串，只含 xy 两种元素，同时只能互相之间交换如 s1(i)和 s2(j)，不能自己之间交换如 s1(i)和 s1(j)，最终，让两字符串相等，描述算法思想输出使 s1 与 s2 相同的最少交换次数，如果不能相同输出-1.

例：xx 与 yy，得到序列 xy 与 xy，输出为 1。

xx 与 xy，输出为－1。

算法：

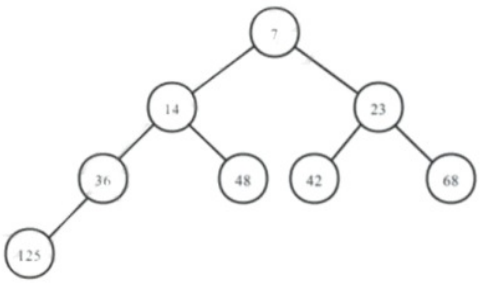
1、若两个字符串长度不同，算法终止返回-1.

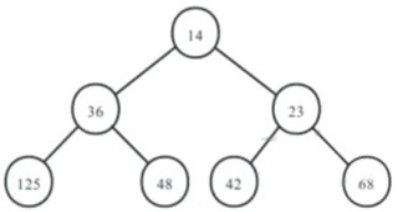
2、初始化变量 xy=0,yx=0，分别记录两个字符串中的 x-y 与 y-x 匹配对的数量.

3、For each pair char1,char2 in str1,str2

```
if char1==char2
continue;
else if char1='x'
xy++;
else
yx++;
if (xy+yx)%2==1
return -1;
return (xy+1)/2+(yx+1)/2
```


3、一维数组存储小顶堆 7,14,64,32,24,68,89,47,38,125，求删除最小元素后的一维数组存储结果，再调整为堆。





数组 14，36，23，125，48，42，68.

4 有 n 个娃娃，每个小朋友一次可以拿 1 或 2 个，算出有多少种情况。
解：
定义 T(n)表示 n 个娃娃的分配方案数。
建立递推式如下

$$T(n)=\begin{cases}1, n=1\\ 2, n=2\\ T(n-1)+T(n-2), otherwise\end{cases}$$

5给了一个表，序号有序，顾客号都是四位数，顾客号无序，问能不能用顾客号二分搜索和原因。
解：不能，不是有序。

序号	数量	顾客号	时间
1	313	3424	2021/12/1
2	12	1231	2012/11/31
3	313	6543	2012/1/12

6、有 n 个箱子初始都是 k 空间，向里面放入体积小于 k，且不可分割的物品，为达到箱子空间的最大利用率，每次物品均放入剩余空间尽可能接近物体体积的箱子。问怎样安排能最快找到最适合的箱子.
解：按二叉排序树，存储，查找最佳的体积容量，大于 key 的最小元素。
算法：
1、统计所有箱子的剩余空间体积（初始 a[n]=k）.
2、建立二叉排序树，初始化为空。每当使用一个箱子后，就将其剩余容量作为结点插入二叉排序中。
3、每次寻找最佳箱子，采用二叉树的查找算法，key 为待装入物品的体积。若找到了，删除结点，更新结点数据，重新插入二叉树中；若二叉树未找到，则新建一个结点，重新赋值，插入二叉树中。

7.人与人之间都有一定关系，现限定人之间只有喜欢的关系，A 喜欢 B，则 B 的被喜欢程度加 1，给定一定关系，找出被喜欢程度最大的那个人，如有相同被喜欢成都的人，按字典次序输出，如"A B C"，不能出现"A→B→C→A"的环形喜欢，写出基本实现思想.
答：计算每个顶点的入度，按入度递减输出，且存在拓扑排序。

四、算法题

1、给定一个不递减的带头结点的单链表，讲这些结点分化为不含相同结点的几个递增序列，重复次数最多 10 次，把它分成若干没有重复数字的递增序列，如 1 2 2 3 3 3 4 4 5 5 6 分成

1 2 3 4 5 6

2 3 4 5

3

```
#include<stdio.h>
#include<stdlib.h>
typedef struct LNode{
        int data;
        LNode* next;
}LNode,*LinkList;

LinkList Partion(LinkList L0) {
        LinkList L;
        LNode A[10], R[10];
        L = A;
        int pre = -10000;
        LinkList p;
        p = L0->next;
        for (int i = 0; i < 10; i++) {
                R[i].next = (L+i);
        }
        LNode* new_p;
        new_p = (LNode*)malloc(sizeof(LNode));
        if (p != NULL) {
                new_p->data = p->data;
                R[0].next->next = new_p;
                R[0].next = new_p;
                pre = p->data;
                p = p->next;
        }
        int k = 1;
        while (p != NULL) {
                if (pre != p->data) {
                        new_p = (LNode*)malloc(sizeof(LNode));
                        new_p->data = p->data;
                        R[0].next->next = new_p;
                        R[0].next = new_p;
                        pre = p->data;
                        p = p->next;
                        k = 1;
                }
                else {
                        new_p = (LNode*)malloc(sizeof(LNode));
                        new_p->data = p->data;
                        R[k].next->next = new_p;
                        R[k].next = new_p;
                        pre = p->data;
                        p = p->next;
                        k++;
                }
        }
for (int i = 0; i < 10; i++) {
                R[i].next->next = NULL;
        }
        return L;
}
void main(){
        LinkList L,p;
        L = (LNode*)malloc(sizeof(LNode));
        p = L;
        LNode* new_i;
        int a[10] = { 1,2,3,4,4,4,5,5,6,6 };
        for (int i = 0; i < 10; i++) {
                new_i = (LNode*)malloc(sizeof(LNode));
                new_i->data = a[i];
                p->next = new_i;
                p = p->next;
        }
        p->next = NULL;
        p = L->next;
        for (int i = 0; i < 10; i++) {
                printf("%d", p->data);
                p = p->next;
        }
        printf("\n");
        p = L->next;
        LinkList L_new;
        L_new=Partion(L);
for (int i = 0; i < 10; i++) {
        p = L_new[i].next;
        while (p != NULL) {
                printf("%d", p->data);
                p = p->next;
        }
        printf("\n");
}
}
```

2、给了二叉树的数据结构，写判断是否为平衡二叉树。

解：

```
void Judge_AVL(BiTtree bt, int& balance, int& h) {
        int bl = 0, br = 0, hl = 0, hr = 0;
        if (bt == NULL) {
                h = 0;
                balance = 1;
        }
        else if (bt->lchild == NULL && bt->rchild == NULL) {
                h = 1;
                balance = 1;
        }
        else {
                Judge_AVL(bt->lchild, bl, hl);
                Judge_AVL(bt->rchild, br, hr);
                h = (hl > hr ? hl : hr) + 1;
                if (abs(hl - hr) < 2)
                        balance = bl && br;
                else
                        balance = 0;
        }
}
```