2021－2022 学年第二学期湖南师范大学
信息科学与工程学院计算机科学与技术专业 2019 级
《算法分析与设计》课程期末考试答案与评分标准

课程代码：12160163　考核方式：开卷　考试时量：120 分钟　　试卷类型：A

一、单项选择题（每小题 3 分，共 30 分）：

1	2	3	4	5	6	7	8	9	10
B	C	D	A	C	D	A	B	D	A

二、判断题（每小题 2 分，共 16 分）

11	12	13	14	15	16	17	18
×	×	×	√	√	×	√	√

三、简答题（每小题 10 分，共 30 分）

19．(1)当 g 不是连通图时，BFS 不能遍历所有顶点。－－－－2 分

补充主程序：

```
void BFSmain(Graph g) //图结构为邻接矩阵或邻接表。
{  for each v in g
         visited[v] = 0;     //所有顶点访问标记初始化为0
   for each v in g           //没有被访问的顶点作为起始点
         BFS(g,v);
}  ―――――――――――――――――――――――――――――――3分
```

（2）设置前驱数组置初值为 0；① prev[v]=-1；②if(w==v) return findpath(prev,v); ③prev[u]=w; ――――――――――――――――――――――――――――5 分

20.（1）贪心选择策略：所有活动按结束时间递增排序，即 $f_1\le \dots \le f_n$. 第一次选择第一个活动 $<s_1,f_1>$，以后每次在当前活动完成后，选择一个可以开始且最早结束的活动$<s_k,f_k>$。S_k>=当前活动完成时间，保证相容。－－－－－－－－－－－－－－－－4 分

（2）该策略能得到问题的最优解，用数学归纳法证明。－－－－－－6 分

当 $k = 1$ 时， 证明存在包含活动 1 的最优解。任取最优解 A， A 中的活动按截止时间递增排列。如果 A 的第一个活动为 $j \ne 1$，令 $A'=(A-\{ j \})\cup\{1\}$， 由于 $f_1 \le f_j$，则 A'也是最优解且含有 1.

假设命题对 $k-1$ 为真，证明对 k 也为真。即当算法执行到第 $k-1$ 步时，存在最优解 A 包含活动 $i_1=1, i_2, \dots, i_{k-1}$，不妨设 $A = \{i_1=1, i_2, \dots , i_{k-1} \}\cup B$，其中 B 的活动来自于剩余活动集合 $S'=\{ i | i\in S, s_i \ge f_{k-1}\}$. 可以证明 B 是 S'的最优解。

若不然，S'的最优解为 B^*，B^*的活动比 B 多，那么 $B^*\cup\{1, i_2, \dots , i_{k-1}\}$是 S 的最优

解，且比 A 的活动多，与 A 是最优解矛盾。矛盾说明 B 是 S' 的最优解，根据归纳法的第一步知，存在 S' 的最优解 B' 含有 S' 中的第一个活动，即 i_k，且 $|B'|=|B|$，于是

$$\{i_1=1, i_2, \dots, i_{k-1}\}\cup B'=\{i_1=1, i_2, \dots, i_{k-1}, i_k\}\cup(B'-\{i_k\})$$

也是原问题的最优解.

21.（1）采用的是回溯法策略，属于子集树空间搜索。—————————4 分

（2）可以求解 0-1 背包问题，设 n 个物品的重量和价值分别是数组 w[1..n]和 v[1…n]，背包容量为 B. 又设当前得到的最优值为 optV，迭代搜索的合法性判定条件如下：

```
Bool legal(t, i) //若合法返回 True，否则返回 False.————————6 分
{  if (i)  //当 i=1 时添加物品 t
   {    rw=B-sum(x[j]*w[j]:j=1,…,t-1);   //求背包剩余容量
        if(w[t]<=rw) return True;
   }
   else  //当 i=0 时不添加物品 t
   {  cv= sum(x[j]*v[j]:j=1,…,t-1)+ sum(v[j]:j=t+1,…,n); //当前背包价值加上剩余物品总价值
      if(cv>optV) return True;
   }
   return False;
}
```

四、算法题（每小题 12 分，共 24 分）

22.（1）findMedian(A,B,lA,rA,lB,rB) ————————8 分

```
{   if(lA==rA && lB==rB) return A[lA];
    midA = int((lA+rA)/2);
    midB = int((lB+rB)/2);
    if (A[midA]==B[midB]) return A[midA];
    if (A[midA]<B[midB])
         return findMedian(A,B,midA,rA,lB,midB);
    else
         return findMedian(A,B,lA,midA,midB,rB);
}
```

（2）设问题<A,B,0,n-1,0,n-1 >的时间为 T(n)，则有

$T(1)=1$; $T(n)=T(n/2)+1$ $(n>1)$

解得 T(n)=O(logn). 空间复杂度 O(n)————————————4 分

23. （1）子问题 $p(i, b)$定义为前 i 个钱币兑换整钱 b，使得需要的钱币数最少。

(2) 设最优函数为 dp[i][b]，表示子问题 $p(i, b)$的最少钱币数。递推关系如下：

dp[i][0]=0, i=1,2,…,n;

dp[0][b]=B+1, b=0,1,…,B;

dp[i][b]=min{dp[i-1][b-k*c_i] : k=0,1,2,…,int(b/c_i)}+k; ————3 分

(3) 设钱币数组为 c[1…n], 辅助解向量为 fk。fk[i][b]为 dp[i][b]取得最小值时零钱 c_i 的个数（即 k 值）。求解 dp[i][b]和 fk[i][b]的算法描述如下：

```
solveDP(int[] c,int B, int[][] dp, int[][] fk)
{  for (b=0;b<=B;b++)  dp[0][b]=B+1;
   for (i=1;i<=n;i++)  dp[i][0]=0;
      for (b=1;b<=B;b++)
      {   dp[i][b]= dp[i-1][b];
          for (k=1;k<=int(b/c[i]);k++)
               if(dp[i][b]> k+dp[i-1][b-k*c[i]])
               {    dp[i][b]= k+dp[i-1][b-k*c[i]];
                    fk[i][b]=k;
               }
      }
}   ————————————————————3 分
```

(4) 根据 dp 和 fk 求 p(n,B)的解，从[n][B]出发，算法如下：

```
Traceback(fk,x) //x 是解向量。
i=n;
b=B;
while(i>=1)
{   x[i]=fk[i][b];
    b -= x[i]*c[i];
    i--
}  ——————————————————————3 分
```

(5) 算法是三重循环，花费时间不超过 O(nB^2)，两个二维数组和一个一维数组，空间复杂度为 O(nB). ————————————————————3 分