中级软件设计师下午试题模拟58

试题一

1、 使用说明中的词语，给出下图中的数据存储D1～D5的名称。

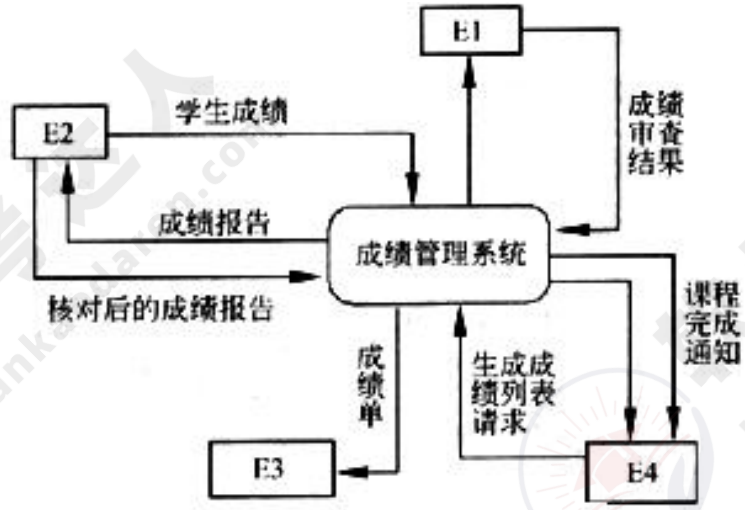


图 1-1 顶层数据流图

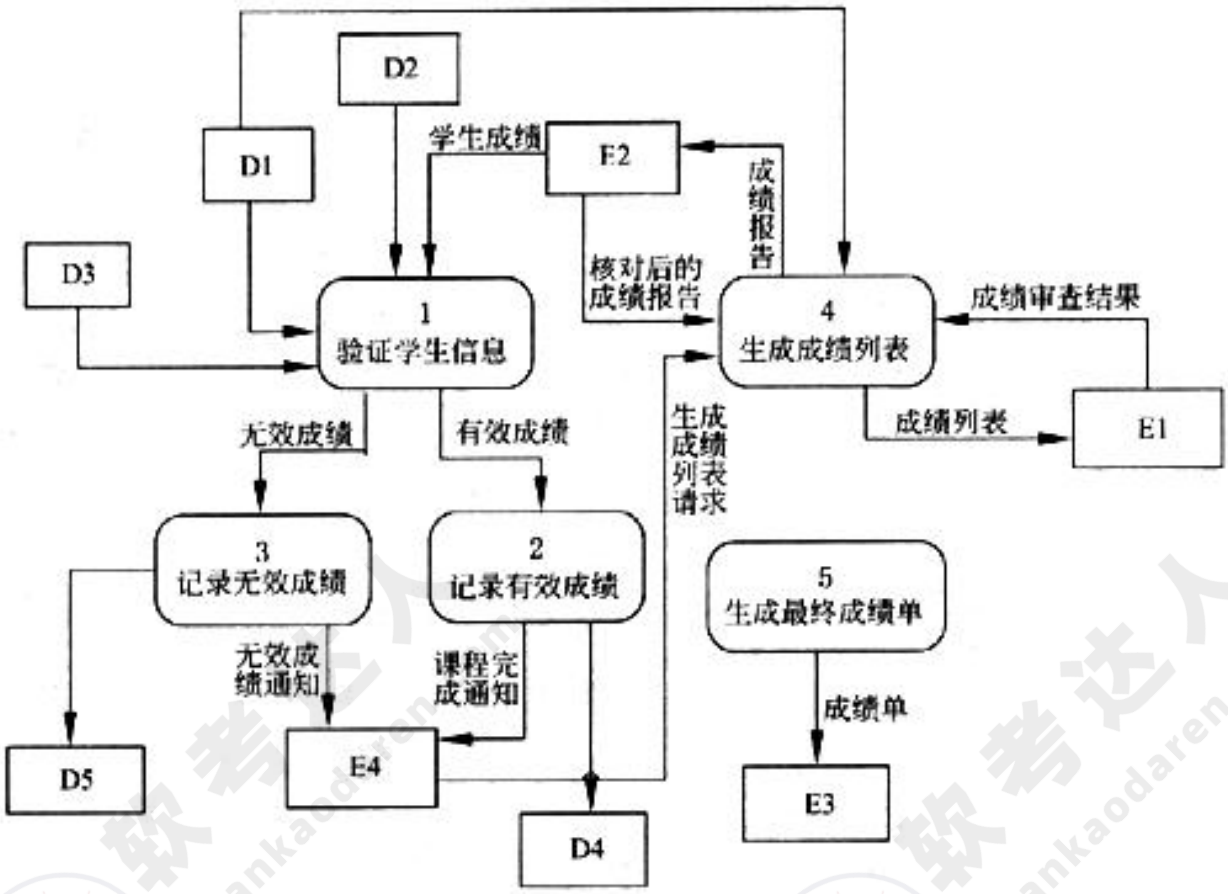


图 1-2 0 层数据流图

试题二

[说明]

一个描述学校的部分关系模式的结果描述如下：

1．一个系有若干学生，但一个学生只能在一个系；

2．一个系只有一名主任；

3．一个学生可以选修多门课程，每门课程有若干学生选修；

4．每个学生所学的每门课程都有一个成绩；

5．“学生”和“课程表”及“选课表”的关系示例分别如表1、表2、表3所示。

Student(学生表)的字段按顺序为学号(Sno)、姓名(Sname)、性别(Ssex)、年龄(Sage)、所属院系(Sdept)、系主任(Smaster)；

Course(课程表)的字段按顺序为课程编号(Cno)、课程名(Cname)、先行课程(Cpno)、课程学分 (Ccredit)；

SC(选课表)的字段按顺序为学号(Sno)、课程号(Cno)、成绩(Grade)。

各表的记录如下：

表1 Student

Sno	Sname	Ssex	Sage	Sdept	Smaster
95001	李勇	男	20	CS	王平
95002	刘晨	女	19	IS	周言
95003	王明	女	18	MA	展评
95004	张立	男	19	IS	周言

表2 Course

Cno	Cname	Cpno	Ceredit
1	数据库	5	4
2	数学		2
3	信息系统	1	4
4	操作系统	6	3
5	数据结构	7	4
6	数据处理		2
7	PASCAL	6	4

表3 SC

Sno	Cno	Grade
95001	1	92
95001	2	85
95001	3	88
95002	2	90
95003	3	80

2、 [问题1]

试分析该关系模式中的函数依赖，并指出关系模式的候地选码。

3、 [问题2]

如下的SQL语句是检索“信息系(IS)和计算机科学系(CS)的学生的姓名和性别”的不完整语句，请在空缺处填入正确的内容。

SELECT___(1)___

FROM___(2)___

WHERE___(3)___

4、 [问题3]

如下的SQL语句是检索“每个学生及其选修的课程名和成绩”的不完整语句，请在空缺处填入正确的内容。

SELEC___(1)___

FROM___(2)___

WHERE___(3)___

试题三

[说明]

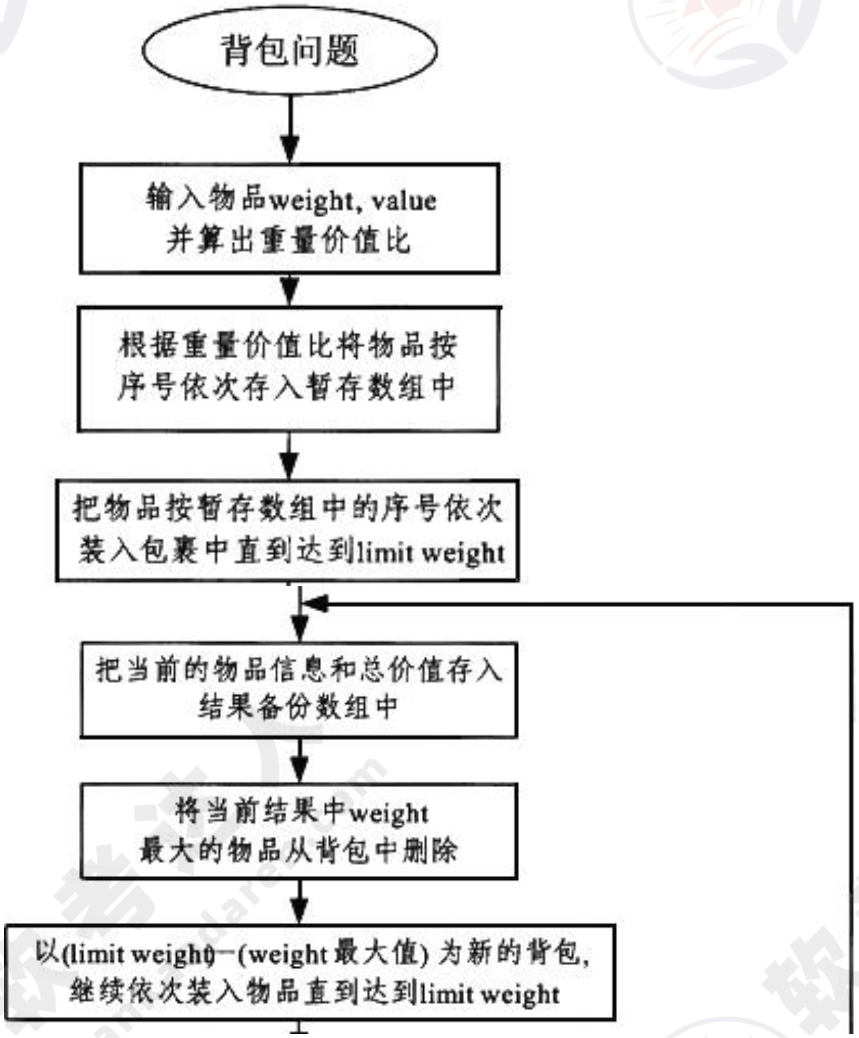
背包问题就是有不同价值、不同重量的物品n件，求从这n件物品中选取一部分物品的选择方案，使选中物品的总重量不超过指定的限制重量，而且选中物品的价值之和为最大。

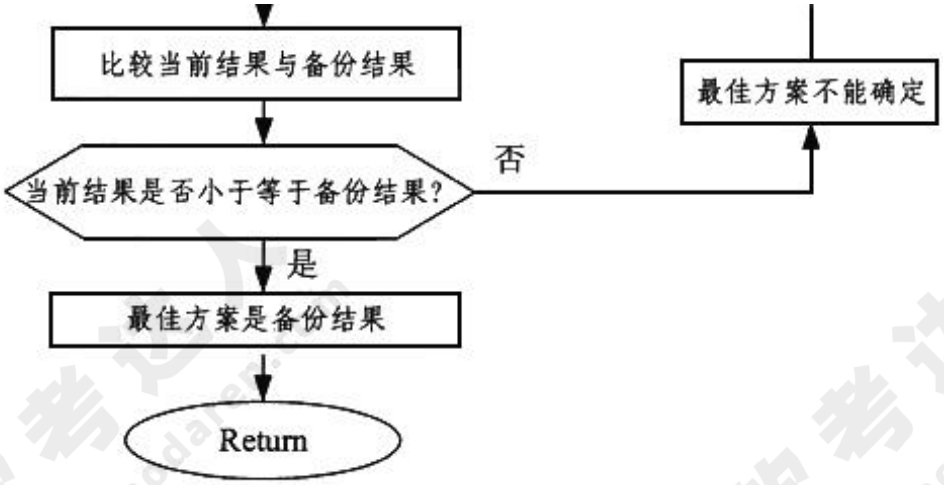
背包问题是一个典型的NP完全难题。对该问题求解方法的研究无论是在理论上，还是在实践中都具有一定的意义。如管理中的资源分配、投资决策、装载问题等均可建模为背包问题。

常用的背包问题求解方法很多，但本题中采用了一种新的算法来求解背包问题。该算法思想为：首先要对物品进行价重比排序，然后按价重比从大到小依次装进包裹。这种方法并不能找到最佳的方案，因为有某些特殊情况存在，但只要把包中重量最大的物品取出，继续装入，直到达到limitweight，这时的物品就是limit weight的最大价值。这种算法不需要逐个进行试探，所以在数据非常大时，执行效率主要由排序的时间复杂度决定。该算法的流程图为下图。

仔细阅读程序说明和C程序流程图及源码，回答问题1和问题2。

[流程图]





[程序说明]

struct Thing：物品结构

typedef struct Bag：背包结构类型

input 10：将物品按序号依次存入数组函数

inbag 10：物品按物价比入包函数

init 10：初始化函数

sort 10：对物品按价格重量比排序函数

outbag 10：取出包中weiht最大的物品函数

print 10：最佳方案输出函数

[C程序]

```
#define N 255
struct Thing {
        double weight;
        double value;
        double dens;
     }thing[N];
typedef stmct Bag {
        Thing thing [N];
        double weighttmp;
        double sumvalue;
    }bag,best;
inbag 10
{
do{
    bag.thing[i]=thing[i]
    ___5___
    ___6___
    i++;
}while (___7___)
}
init 10
{
for (inti=0; i<N; i++)
{
    input (thing[i].weight, thing [i].value)
    thing [i].dens=thing[i].value/thing [i].weight;
};
}
main 10
```

```
{
init 10;
sort 10;
inbag 10;
do {
   best=bag;    //把包中物品放入暂存数组
   outbag 10;   //取出包中weight最大的物品
   ___8___
    }while (___9___)
print (best);   //输出temp因为是最佳方案
}
```

5、 [问题1]

根据程序说明及流程图、部分C源码，充分理解算法思想，填入__(n)__处。

6、 [问题2]

求解"背包问题"常用的方法有哪几种?各有什么样的特点?

试题四

7、 【说明】设某城市有n个车站，并有m条公交线路连接这些车站，设这些公交车都是单向的，这n个车站被顺序编号为0至n-1。本程序，输入该城市的公交线路数、车站个数，以及各公交线路上的各站编号，求得从站0出发乘公交车至站n-1的最少换车次数。

程序利用输入信息构建一张有向图G(用邻接矩阵g表示)，有向图的顶点是车站，若有某条公交线路经i站到达j站，就在顶点i到顶点j之间设置一条权为1的有向边<i,j>。如果这样，从站点x至站点y的最少上车次数便对应图G中从点x到点y的最短路径长度。而程序要求的换车次数就是上车次数减1。

```
#include <stdio.h>
#define    M     20
#define    N     50
int a[N+1];      /*用于存放一条线路上的各站编号*/
int g[N][N];     /*严存储对应的邻接矩阵*/
int dist[N];     /*严存储站0到各站的最短路径*/
int m, n;
void buildG()
{   int i, j, k, sc, dd
    printf("输入公交线路数，公交站数\n");
    scanf("%d%d", &m, &n);
    for (i=0; i<n; i++)      /*邻接矩阵清0*/
     for(j=0; j<n; j++)
        g[i][j]=0;
for(i=0; i<m; i++)
{    printf("沿第%d条公交线路的各站编号(0<=编号<=%d,-1结束):\n)",i+1,n-1);
     sc=0;       /* 当前线路站计数器*/
     while(1)
     {   scanf("%d", &dd);
         if(dd==-1)break;
         if(dd>=0 && dd<n) __(1)__;
     }
     a[sc]=-1;
     for(k=1; a[k]>=0; k++)     /*处理第i+1条公交线路*/
          for(j=0; j<k; j++)
```

```
                    g__(2)__=1;
    }
}
int minLen()
{   int j, k;
    for(j=0; j<n; j++)
         dist[j]=g[0][j];
    dist[0]=1;
    do{
         for(k=-1, j=0; j<n; j++)        /*找下一个最少上车次数的站*/
              if(dist[j]>0 &&(k==-1 | |dist[j]<dist[k]))
                   k=j;
         if(k<0 | |k==n-1)
              break;
         dist[k]=-dist[k];          /*设置k站已求得上车次数的标记*/
         for (j=1; j<n; j++)       /*调整经过k站能到达的其余各站的上车次数*/
              if(__(3)__&& (dist[j]=0 | |-dist[k]+1<dist[j]))
                   dist[j]= __(4)__;
    }while(1);
    j=dist[n-1];
    return __(5)__;
}
void main()
{   int t;
    buildG();
    if((t=minLen())<0)
         printf("无解!\n");
else
     printf("从0号站到%d站需换车%d次\n", n-1, t);
}
```

试题五

8、 阅读下列程序说明和C程序，将应填入__(n)__处的字句写在答卷纸的对应栏内。

【程序说明】

该程序定义了两个子函数strsort和strmerge。它们分别实现了将一个字符串按字母顺序排序和将两个字符串合并排序，并删去相同字符。在主函数里，先输入两个字符串s1和s2，然后调用strsort函数对它们分别排序，然后调用strmerge函数将s1和s2合并，将合并后的字符串赋给字符串s3，最后输出字符串s3。

【程序】

```
 #include <stdio.h>
void strmerge(char *a,char *b,char *c)          //将字符串a,b合并到字符串c
{
  char t,*w;
  W=c;
   while(__(1)__)
  {
```

```
//找到字符串a,b当前字符中较小的字符
  if(*a<*b)
   {
     t=-*a,
     (2)
   }
  else if(*a>*b)
   {
     t=*b;
     (3)
   }
  else                          //字符串a,b 当前字符相等
   {
     t=-*a;
     a-H-;
     b-H-;
   }
  if( (4) )        //开始，可直接赋值
     *w=t;
  else if(t!=*w)
  //如果a,b中较小的当前字符与c中当前字符不相等，才赋值
  (5)
}
if(*a!='\0')                    //如果字符串a还没有结束，则将a的剩余部分赋给c
while(*a!='\0')
  if(*a!=*w)
  {
     *(++w)=*a;
     a++;
   }
  else
  (6)
if(*b!=",'\0')                  //如果字符串b 还没有结束，则将 b 的剩余部分赋
给 c
while(*b !='\0')
  if(*b!=*w)
   {
     *(++w)=*b;
     b++;
    }
  else
     b++;
     (7)
 }
 void strsort(char *s)                  //将字符串 s 中的字符排序
{
   int i,j,n;
   char t,*w;
   w=s;
   for(n=0;*w!='\0';n++)                //得到字符串长度 n
     w++;
```

```
    for(i=O;i<n-1;i++)                    //对字符串 s 进行排序，按字母先后顺序
      forO=i+ 1 ;j<n;j++)
        if( (8)
          {
            t=s[i];
            s[i]=s[j];
            (9)
          }
    }
  void mainO
  {
    char s1 [100],s2[100],s3[100];
    prinff("\nlPlease input the first string:");
    scanfC("% s",s1 );
    prinff("\nPlease input the second string:");
    scanf("%s",s2);
    strsort(s1);                          //将字符串s1 排序
    strson(s2);                           //将字符串 s2 排序
    prinff("%s\n',s1);
    printfC % sW',s2);
    s3[0]='\O';                           //字符串 s3 的第一个字符先置'\0'结束标志
    (10) ;          //将s1和s2合并，按照字母顺序排列，
     prinff("%s",s3);
  }
```

答案：

试题一

1、D1：学生信息文件；D2：课程单元信息文件：D3：课程信息文件；D4：课程成绩文件；
D5：无效成绩文件。
注：D2和D3的答案可以互换。

试题二

2、在该关系模式中，存在以下函数依赖：
学号→姓名　　学号→所在系　　所在系→系主任
(学号，课程名)→成绩
系主任传递的依赖学号；
该关系模式的候选码为(学号，课程名)；
姓名、所在系部分依赖候选码。

[解析]
试题二
本题考查的是基础知识，考生如果掌握对关系模式和SQL语言的相关知识可得出答案。

3、(1)Sname, Ssex
(2)Student
(3)Sdept IN('IS','CS')

4、(1)Student.Sno,Sname,Course.Cname,SC.Grade
(2)Student,SC,Course

(3)Student.Sno=SC.Sno and SC.Cno=Course.Cno;

试题三

5、bag.weighttmp=bag.weighttmp+thing[i].weight;

(2)bag.sumvalue=bag.sumvalue+thing[i].value;

(3)bag.weighttmp<=weightlimit

(4)inbag ( );

(5)best.sumvalue<bag.sumvalue

[解析] 本题考查的是考生对流程图的阅读能力。本题涉及的算法是背包问题。背包问题求解方法很多，考生首先要理解本题中的新方法，然后对照流程图阅读代码。(1)处应该为物品总重量；(2)处应该为物品总价值；(3)处应该为直到达到极限重量limit weight；(4)处应该为继续装物品；(5)处应该为比较当前结果与备份结果。问题2同样是考查有关基本概念的问题。根据软件设计师考试的趋势，本套题设计上有意识地增加了概念考查部分，希望考生能够加强对基本概念的理解与训练。

6、"背包问题"求解方法主要是一些启发式算法，如贪婪算法、递归算法等。应用递归算法的目的是穷举所有可能的解，从中选出最佳解。这种解法实际上是穷举了所有的可能，只是加了一些限制。如果所求的数据很大，这种算法的效率就不是很高，甚至是不可实现的。贪婪法不用穷举且速度快，但用贪婪法却不一定能找到最优解。由于贪婪法所得到的解与最优解存在很大的差距，当要求较高时，就会成为贪婪法致命的且无法挽救的缺陷。

试题四

7、[解析]

(1)a[sc++]=dd

将dd赋值给当前线路的a[sc]，并同时将当前线路站计数器加1。

(2)[a[J][a[k]]

将a[j]和a[k]之间置为通路1。

(3)dist[j]>=0 && g[k][j]==1

若dist[j]并且k和j之间有通路，或-dist[k]+1<dist[j]，也就是经过k对于j来说上车次数更少，则调整经过k站能到达的其余各站的上车次数。

(4)-dist[k]+1

让dist[j]取经过k的更少上车次数-dist[k]+1。

(5)k<0?-1: j-1

若k小于0，表示无解；否则返回j-1也就是上车次数减1。

试题五

8、(1)(*a!='\0')&&(*b!='\0')

(2)a++

(3)b++

(4)*w=='\0'

(5)*(++w)=t

(6)a++

(7)*(++w)='\0'

(8)s[i]>s[j]

(9)s[j]=t

(10)strmerge(s1,s2,s3)

[分析]

根据题意，对字符串的处理分为三步：第一步是从键盘上输入两个字符串：第二步是将两个字符串分别排序；第三步是将字符串合并；第四步是显示处理结果。

第一步和第四步容易实现，关键是第二步和第三步的处理，下面分别加以说明。

字符串排序是指将一个字符串中各个字符按照ASCII码值的大小排序。例如，字符串"Beijing"由小到大的排序结果应该是："Bejiign"。排序算法很多，第二个例子，我们就要介绍快速排序算法。在这里使用简单的冒泡排序算法：即将字符串中的每一个字符一个个进行比较，找出最小的字符，然后再在剩下的字符中找最小的字符。例如，字符"Beijing"的排序过程如下：

第一次将字符"Beijing"中的每一个字符：'B'、'e'、'i'、'j'、'i'、'n'、'g'进行比

较，找到最小的字符'B'。

第二次在剩下的字符'e'、'i'、'j'、'n'、'g'中，找到最小的字符'e'。

第三次在剩下的字符'i'、'j'、'i'、'n'、'g'中，找到最小的字符'j'。

第三步是合并字符串，合并后的字符串仍然由小到大排序。由于待合并的两个字符串已经排好序。假定两个排好序的字符串分别为A和B，合并后的字符串为巴要使待合并后的字符串仍然由小到大排序，可采取下述步骤：

1．从前往后取A中的字符，并按从前往后的顺序与B中的字符比较，若A中的字符较小，则将该字符存入C，并移到A的下一个字符，继续与B中的字符比较。

2．若A中的字符较大，则将B中的字符存入C，并移到B的下一个字符，继续与A中的字符比较。

3．若A与B中的字符相等，则将A或B中的字符存入C并将A和B均移到下一个字符。

4．若A或B字符串到达末尾，则将B或A的剩余部分加到字符串C中。

需要注意的是：A、B和C三个字符串均可以用字符数组来表示，C数组的长度不能小于A、B两数组的长度之和。另外，判别字符串是否结尾的方法是：从A或B中取出的字符是否为'\0'，所有字符串都是以'\0'结尾的。