

目录

1. 需求分析.....	1
2. 项目亮点.....	1
3. 概要设计.....	1
4. 详细设计.....	1
4.1 创建有向图.....	1
4.2 拓扑排序.....	2
4.3 课程安排方案.....	3
4.3.1 均匀安排课程.....	3
4.3.2 尽快安排课程.....	4
5. 用户手册.....	5
5.1 测试数据.....	5
5.2 测试结果.....	6
5.3 错误案例.....	7
6. 心得体会.....	7
7. 附录	8

1. 需求分析

- (1) 每个专业开设的课程都是确定的，而且课程在开设之间的安排必须满足先修关系。每门课程有哪些先修课程是确定的，可以有多门，也可以没有，每门课占一个学期。试在这样的前提下设计一个教学计划编制程序。
- (2) 输入参数包括：学期总数（不超过 12），一学期的学分上限，每门课的课程号、学分和直接先修课的课程号，课程总数不超过 100。如果输入的先修课程号不在该专业开设的课程序列中，则作为错误处理。
- (3) 允许用户指定下列两种编排策略之一：一是使学生在各学期中的学习负担尽量均匀；二是使课程尽可能地集中在前几个学期中。
- (4) 若根据给定的条件问题无解，则报告适当的信息；否则将教学计划输出到用户指定的文件中，并设计计划的表格格式。
- (6) 产生多种不同的方案，并使方案之间的差异尽可能地大。

2. 项目亮点

如果有向图无回路，用户可以自行选择生成 2-10 种差异很大的拓扑排序序列；否则系统提示无法设计教学计划。

3. 概要设计

拓扑排序是一个有向无环图的所有顶点的线性序列，每个有向无环图可以产生多种不同的序列。设计使用邻接表构建有向图，为产生差异较大的序列，这里将入度为 0 的顶点存入线性表，首先正序和逆序输出线性表，即为两种拓扑序列。然后利用线性表的特点和随机函数从表中随机输出顶点，即为更多的拓扑序列。

4. 详细设计

4.1 创建有向图

如果读取文件时同时读取课程编号和先决条件，当课程编号顶点未建立时，无法形成对应的弧，如 C6 的先决条件是 C11，但此时未建立顶点 C11，无法形成边。

因此首先读取用户指定的文件，第一行为学期总数、每学期最大学分上限、课程总数；第二行为所有课程编号，先建立顶点，再读取先决条件；之后的每一行为对应的学分和先决条件。在读取结束后，将生成对应的有向图。

Listing 1: Create

```
1 int Create(AlGraph* G) //创建有向图
2 {
3     int i,j,s,count;
4     char a,name[30],str[4];
```

```

5 FILE *fp;
6 printf("<<<---教学计划编制系统--->>>\n");
7 A:printf("请输入课程表文件名:");
8 fflush(stdin);gets(name);strcat(name, ".txt");
9 if((fp=fopen(name, "r"))==NULL) {printf("该文件夹下无%s,请重新输入!\n",name);goto A;}
10 G->message=(Term *)malloc(sizeof(Term));
11 G->courses=(ClassNode *)malloc(sizeof(ClassNode)*G->vexNum);
12 fscanf(fp, "%d%d\n\n", &G->message->termNum, &G->message->maxCredit, &G->vexNum);
13 if(G->message->termNum>12 || G->message->termNum<1 || G->message->maxCredit<1
14 || G->vexNum>100 || G->vexNum<1)
15     {printf("输入超出范围!\n");return 0;}
16 for(i=0; i<G->vexNum; i++) //初始化
17 {
18     fscanf(fp, "%s", G->courses[i].classNum);
19     G->courses[i].firstEdge=NULL;
20     G->courses[i].inDegree=0;
21 }
22 for(i=0; i<G->vexNum; i++)
23 {
24     fscanf(fp, "%d", &G->courses[i].credit); //读取课程编号和学分
25     while(fgetc(fp)!='\n') //根据先决条件建立邻接表结点
26     {
27         fscanf(fp, "%s", str);
28         s=(strlen(str)==2)?str[1]-'1':(str[1]-'0')*10+str[2]-'1'; //课程字符串转数字
29         if(s<0 || s>G->vexNum) {printf("%s输入错误!\n", G->courses[i].classNum);return 0;}
30         EdgeNode *p=(EdgeNode*)malloc(sizeof(EdgeNode)); //更新邻接表结点
31         p->adjVex=i;
32         p->next=G->courses[s].firstEdge;
33         G->courses[s].firstEdge=p;
34         G->arcNum++;
35     }
36 }
37 fclose(fp);
38 for(i=0; i<G->vexNum; i++)
39 {
40     for(EdgeNode *p=G->courses[i].firstEdge; p!=NULL; p=p->next)
41         G->courses[p->adjVex].inDegree++;
42 }
43 return 1;
44 }

```

4.2 拓扑排序

利用有向图和线性表产生拓扑序列，将入度为0的点存入线性表，相关的邻接点入度减一，然后按照指定的规则从线性表中输出一个顶点，重复以上过程至线性表为空。

如果输出个数不为顶点个数，则说明有向图中存在环，提示用户无法生成拓扑序列。
用户自行指定使用其中一种序列作为教学计划顺序。

Listing 2: TopSort

```

1  int TopSort(ClassNode g[],int n,ClassNode *temp,int type) //拓扑排序
2  {
3      int i,j,k,m=0,gd[n],x;
4      for(i=0;i<n;i++) gd[i]=g[i].inDegree;
5      List degree;
6      degree.length=0;
7      EdgeNode* p;
8      if(type==1||type>=a/2) for(i=n-1;i>=0;i--) {if(gd[i]==0) Add(&degree,i);}
9      else for(i=0;i<n;i++) {if(gd[i]==0) Add(&degree,i);}
10     printf("\n%d种拓扑排序结果为:",type);
11     while(degree.length!=0)
12     {
13         if(type<3) x=0;
14         else x=rand()%degree.length;
15         j=degree.data[x]; //输出链表的某个元素
16         Remove(&degree,x+1);
17         printf("%s ",g[j].classNum);
18         temp[m++]=g[j];
19         for(p=g[j].firstEdge;p;p=p->next) //删除顶点j的所有出边
20         {
21             k=p->adjVex;
22             if(!(--gd[k])) //若顶点k入度为零则入链表
23             {
24                 if(type==2) Insert(&degree,1,k);
25                 else if(type<a/2) Add(&degree,k);
26                 else Insert(&degree,1,k);
27             }
28         }
29     }
30     if(m<n) {printf("AOV网有回路,无法设计教学计划!");return 0;}
31     return 1;
32 }

```

4.3 课程安排方案

4.3.1 均匀安排课程

Listing 3: SortA

```

1  void SortA(ClassNode* t,Term* s,int classNum) //均匀安排课程
2  {

```

```

3     char name[30];
4     printf("请输入教学计划编制文件名:");
5     fflush(stdin);gets(name);strcat(name, ".csv");
6     FILE *fp=fopen(name, "w");
7     printf("均匀安排课程的方案如下:");
8     fprintf(fp, "学期, \t, 课程");
9     int i, j, b, c = 0; //用于输出课程信息
10    for(i = 0; i < s->termNum; i++)
11    {
12        b = 0; //累计每学期学分
13        printf("\n第%d个学期的课程为:", i + 1);
14        fprintf(fp, "\n第%d学期, \t, ", i + 1);
15        for(j = 0; j < classNum / s->termNum; j++)
16        {
17            if(b + t[c].credit <= s->maxCredit) //判断是否超过最大学分
18            {
19                if(c == classNum) break;
20                printf("%s ", t[c].classNum);
21                fprintf(fp, "%s, ", t[c].classNum);
22                b = b + t[c].credit; //学分累计
23                c++; //指向下一课程
24            }
25        }
26        if(i < classNum % s->termNum) //加入平均后多余的课程
27        {
28            if(c == classNum) break;
29            printf("%s ", t[c].classNum);
30            fprintf(fp, "%s, ", t[c].classNum);
31            b = b + t[c].credit;
32            c++;
33        }
34    }
35    fclose(fp);
36 }

```

4.3.2 尽快安排课程

Listing 4: SortB

```

1 void SortB(ClassNode* t, Term *s, int classNum) //尽快安排课程
2 {
3     char name[30];
4     printf("请输入教学计划编制文件名:");
5     fflush(stdin);gets(name);strcat(name, ".csv");
6     FILE *fp=fopen(name, "w");

```

```
7 printf("尽快安排课程的方案如下:");
8 fprintf(fp,"学期,\t,课程");
9 int i,b,c=0;
10 for(i=0;i<s->termNum;i++)
11 {
12     b=0; //累计每学期学分
13     printf("\n第%d个学期的课程为:",i+1);
14     fprintf(fp, "\n第%d学期,\t,",i+1);
15     while(b+t[c].credit<=s->maxCredit) //判断是否超过最大学分
16     {
17         if (c==classNum)break;
18         printf("%s ",t[c].classNum); //输出课程
19         fprintf(fp,"%s,",t[c].classNum);
20         b=b+t[c].credit; //学分累计
21         c++; //指向下一课程
22     }
23 }
24 fclose(fp);
25 }
```

5. 用户手册

5.1 测试数据

```
1 6 10 12
2 C1 C2 C3 C4 C5 C6 C7 C8 C9 C10 C11 C12
3 2
4 3 C1
5 4 C1 C2
6 3 C1
7 2 C3 C4
8 3 C11
9 4 C5 C6
10 4 C3 C6
11 7
12 5 C9
13 2 C9
14 3 C9 C10 C1
```

5.2 测试结果

```
<<<---教学计划编制系统--->>>
请输入课程表文件名:数据
请输入产生拓扑排序个数:8

1种拓扑排序结果为:C9 C1 C11 C10 C4 C2 C6 C12 C3 C8 C5 C7
2种拓扑排序结果为:C1 C2 C3 C4 C5 C9 C10 C12 C11 C6 C7 C8
3种拓扑排序结果为:C9 C10 C1 C11 C4 C2 C12 C6 C3 C8 C5 C7
4种拓扑排序结果为:C1 C2 C4 C9 C11 C3 C5 C6 C7 C8 C10 C12
5种拓扑排序结果为:C9 C11 C6 C10 C1 C12 C4 C2 C3 C5 C8 C7
6种拓扑排序结果为:C1 C4 C9 C10 C2 C3 C11 C6 C8 C5 C12 C7
7种拓扑排序结果为:C1 C4 C9 C2 C10 C12 C11 C3 C6 C5 C8 C7
8种拓扑排序结果为:C9 C1 C10 C4 C12 C2 C3 C5 C11 C6 C7 C8
请选择拓扑排序:3

1. 均匀安排课程
2. 尽快安排课程
请选择教学计划编制类型:1
请输入教学计划编制文件名:1
均匀安排课程的方案如下:
第1个学期的课程为:C9
第2个学期的课程为:C10 C1
第3个学期的课程为:C11 C4
第4个学期的课程为:C2 C12
第5个学期的课程为:C6 C3
第6个学期的课程为:C8 C5
是否查看另一种编制类型(y|n):y
请输入教学计划编制文件名:2
尽快安排课程的方案如下:
第1个学期的课程为:C9
第2个学期的课程为:C10 C1 C11
第3个学期的课程为:C4 C2 C12
第4个学期的课程为:C6 C3
第5个学期的课程为:C8 C5 C7
第6个学期的课程为:
是否继续操作(y|n):n
>>>-----<<<
期待您的下次使用, 再会!
请按任意键继续. . .
```

图 1: 测试结果

学期		课程	
第1学期		C9	
第2学期		C10	C1
第3学期		C11	C4
第4学期		C2	C12
第5学期		C6	C3
第6学期		C8	C5

图 2: 均匀安排课程

学期		课程		
第1学期		C9		
第2学期		C10	C1	C11
第3学期		C4	C2	C12
第4学期		C6	C3	
第5学期		C8	C5	C7
第6学期				

图 3: 尽快安排课程

5.3 错误案例

```

1 6 10 4
2 C1 C2 C3 C4
3 3
4 4 C3 C1
5 5 C4
6 2 C2

```

```

<<<---教学计划编制系统--->>>
请输入课程表文件名:错误案例
请输入产生拓扑排序个数:3

1种拓扑排序结果为:C1 AOV网有回路,无法设计教学计划!
是否继续操作(y|n):n
>>>-----<<<
期待您的下次使用,再会!
请按任意键继续. . . ■

```

图 4: 错误案例

6. 心得体会

这次课程设计的心得体会通过实践我们的收获如下:

1. 通过这次教学计划编制问题的课程设计, 我们更深刻地了解了拓扑排序的特点与用法。
2. 在实现拓扑排序时, 一开始采用栈的数据结构, 发现无法产生多种序列, 于是选择利用线性表的特点实现。
3. 最初读取文件时采用的方法是同时读取课程编号和先决条件, 将每个顶点的入度打印出来观察发现 C6 的入度为 0, 于是找到了问题根源, 从而解决了问题。

7. 附录

```
1  #include<stdio.h>
2  #include<stdlib.h>
3  #include<string.h>
4
5  typedef struct EdgeNode //邻接表结点
6  {
7      int adjVex;          //邻接点域
8      struct EdgeNode *next; //指向下一个邻边节点的指针域
9  }EdgeNode;
10
11 typedef struct CNode //课程表节点
12 {
13     char classNum[3+1]; //课程编号
14     int credit;          //课程学分
15     int inDegree;        //课程入度
16     EdgeNode *firstEdge; //指向邻接表第一个邻边节点的指针域
17 }ClassNode;
18
19 typedef struct Term //学期信息
20 {
21     int termNum;          //学期数
22     int maxCredit;        //每学期学分上限
23 }Term;
24
25 typedef struct AlGraph //有向图
26 {
27     ClassNode *courses; //邻接表域
28     int vexNum;          //节点数
29     int arcNum;          //边数
30     Term *message;       //学期与课程信息
31 }AlGraph;
32
33 typedef struct List //线性表
34 {
35     int data[100];        //数据
36     int length;           //长度
37 }List;
38 int a;
39
40 int Create(AlGraph* G) //创建有向图
41 {
42     int i,j,s,count;
43     char a,name[30],str[4];
44     FILE *fp;
```

```

45     printf("<<<---教学计划编制系统--->>>\n");
46     A:printf("请输入课程表文件名:");
47     fflush(stdin);gets(name);strcat(name, ".txt");
48     if((fp=fopen(name, "r"))==NULL) {printf("该文件夹下无%s,请重新输入!\n",name);goto A;}
49     G->message=(Term *)malloc(sizeof(Term));
50     G->courses=(ClassNode *)malloc(sizeof(ClassNode)*G->vexNum);
51     fscanf(fp, "%d%d%d\n\n",&G->message->termNum,&G->message->maxCredit,&G->vexNum);
52     if(G->message->termNum>12||G->message->termNum<1||G->message->maxCredit<1||G->vexNum>100||G->vexNum<1)
53         {printf("输入超出范围!\n");return 0;}
54     for(i=0;i<G->vexNum;i++) //初始化
55     {
56         fscanf(fp, "%s", G->courses[i].classNum);
57         G->courses[i].firstEdge=NULL;
58         G->courses[i].inDegree=0;
59     }
60     for(i=0;i<G->vexNum;i++)
61     {
62         fscanf(fp, "%d",&G->courses[i].credit); //读取课程编号和学分
63         while(fgetc(fp)!='\n') //根据先决条件建立邻接表结点
64         {
65             fscanf(fp, "%s", str);
66             s=(strlen(str)==2)?str[1]-'1':(str[1]-'0')*10+str[2]-'1'; //课程字符串转数字
67             if(s<0||s>G->vexNum) {printf("%s输入错误!\n",G->courses[i].classNum);return 0;}
68             EdgeNode *p=(EdgeNode*)malloc(sizeof(EdgeNode)); //更新邻接表结点
69             p->adjVex=i;
70             p->next=G->courses[s].firstEdge;
71             G->courses[s].firstEdge=p;
72             G->arcNum++;
73         }
74     }
75     fclose(fp);
76     for(i=0;i<G->vexNum;i++)
77     {
78         for(EdgeNode *p=G->courses[i].firstEdge;p!=NULL;p=p->next)
79             G->courses[p->adjVex].inDegree++;
80     }
81     return 1;
82 }
83 void Add(List *L,int data)//添加元素
84 {
85     L->data[L->length++]=data;
86 }
87
88 void Remove(List *L,int number)//删除元素
89 {
90     for(int i=number-1;i<L->length;i++)

```

```

91     L->data[i]=L->data[i+1];
92     L->length--;
93 }
94
95 void Insert(List *L,int locate,int object)//插入元素
96 {
97     int i;
98     for(i=L->length;i>=locate-1;i--)
99         L->data[i+1]=L->data[i];
100    L->data[i+1]=object;
101    L->length++;
102 }
103
104 int TopSort(ClassNode g[],int n,ClassNode *temp,int type) //拓扑排序
105 {
106     int i,j,k,m=0,gd[n],x;
107     for(i=0;i<n;i++) gd[i]=g[i].inDegree;
108     List degree;
109     degree.length=0;
110     EdgeNode* p;
111     if(type==1||type>=a/2) for(i=n-1;i>=0;i--) {if(gd[i]==0) Add(&degree,i);}
112     else for(i=0;i<n;i++) {if(gd[i]==0) Add(&degree,i);}
113     printf("\n%d种拓扑排序结果为:",type);
114     while(degree.length!=0)
115     {
116         if(type<3) x=0;
117         else x=rand()%degree.length;
118         j=degree.data[x]; //输出链表的某个元素
119         Remove(&degree,x+1);
120         printf("%s ",g[j].classNum);
121         temp[m++]=g[j];
122         for(p=g[j].firstEdge;p;p=p->next) //删除顶点j的所有出边
123         {
124             k=p->adjVex;
125             if(!(--gd[k])) //若顶点k入度为零则入链表
126             {
127                 if(type==2) Insert(&degree,1,k);
128                 else if(type<a/2) Add(&degree,k);
129                 else Insert(&degree,1,k);
130             }
131         }
132     }
133     if(m<n) {printf("AOV网有回路,无法设计教学计划!");return 0;}
134     return 1;
135 }
136
137 void SortA(ClassNode* t,Term* s,int classNum) //均匀安排课程

```

```

138 {
139     char name[30];
140     printf("请输入教学计划编制文件名:");
141     fflush(stdin); gets(name); strcat(name, ".csv");
142     FILE *fp=fopen(name, "w");
143     printf("均匀安排课程的方案如下:");
144     fprintf(fp, "学期,\t,课程");
145     int i,j,b,c=0; //用于输出课程信息
146     for(i=0;i<s->termNum;i++)
147     {
148         b=0; //累计每学期学分
149         printf("\n第%d个学期的课程为:", i+1);
150         fprintf(fp, "\n第%d学期,\t,", i+1);
151         for(j=0;j<classNum/s->termNum;j++)
152         {
153             if(b+t[c].credit<=s->maxCredit) //判断是否超过最大学分
154             {
155                 if(c==classNum) break;
156                 printf("%s ", t[c].classNum);
157                 fprintf(fp, "%s,", t[c].classNum);
158                 b=b+t[c].credit; //学分累计
159                 c++; //指向下一课程
160             }
161         }
162         if(i<classNum/s->termNum) //加入平均后多余的课程
163         {
164             if(c==classNum) break;
165             printf("%s ", t[c].classNum);
166             fprintf(fp, "%s,", t[c].classNum);
167             b=b+t[c].credit;
168             c++;
169         }
170     }
171     fclose(fp);
172 }
173
174 void SortB(ClassNode* t, Term *s, int classNum) //尽快安排课程
175 {
176     char name[30];
177     printf("请输入教学计划编制文件名:");
178     fflush(stdin); gets(name); strcat(name, ".csv");
179     FILE *fp=fopen(name, "w");
180     printf("尽快安排课程的方案如下:");
181     fprintf(fp, "学期,\t,课程");
182     int i,b,c=0;
183     for(i=0;i<s->termNum;i++)
184     {

```

```

185     b=0; //累计每学期学分
186     printf("\n第%d个学期的课程为:",i+1);
187     fprintf(fp, "\n第%d学期,\t",i+1);
188     while(b+t[c].credit<=s->maxCredit) //判断是否超过最大学分
189     {
190         if(c==classNum) break;
191         printf("%s ",t[c].classNum); //输出课程
192         fprintf(fp,"%s,",t[c].classNum);
193         b=b+t[c].credit; //学分累计
194         c++; //指向下一课程
195     }
196 }
197 fclose(fp);
198 }
199
200 int main()
201 {
202     A:system("color F1");
203     AlGraph G;
204     int i=1,flag,b,x[5],choice;
205     char choose;
206     ClassNode class[11][100];
207     flag=Create(&G);
208     if(flag)
209     {
210         B:printf("请输入产生拓扑排序个数:");scanf("%d",&a);
211         if(a<2||a>10) {printf("输入超出范围,请重新输入!\n");goto B;}
212         x[0]=TopSort(G.courses,G.vexNum,class[0],i);
213         if(x[0])
214         {
215             for(i=1;i<a;i++)
216                 x[i]=TopSort(G.courses,G.vexNum,class[i],i+1);
217             C:printf("\n请选择拓扑排序:");scanf("%d",&b);
218             if(b<1||b>a) {printf("输入错误,请重新输入!");goto C;}
219             b--;
220             printf("\n1.均匀安排课程\n2.尽快安排课程\n");
221             D:printf("请选择教学计划编制类型:");
222             scanf("%d",&choice);
223             if(choice!=0&&choice!=1) {printf("输入错误,请重新输入!\n");goto D;}
224             (choice==1)?SortA(class[b],G.message,G.vexNum):SortB(class[b],G.message,G.vexNum);
225             printf("\n是否查看另一种编制类型(y\n):");fflush(stdin);scanf("%c",&choose);
226             if(choose=='y')
227             {
228                 choice=1-choice;
229                 (choice==1)?SortA(class[b],G.message,G.vexNum):SortB(class[b],G.message,G.vexNum);
230             }
231         }
232     }

```

```
232 }  
233 else printf("录入信息有误!");  
234 printf("\n是否继续操作(y|n):");fflush(stdin);scanf("%c",&choose);  
235 if(choose=='y') {system("cls");goto A;}  
236 printf(">>>-----<<<\n期待您的下次使用,再会!\n");system("pause");  
237 }
```