考试科目名称及代码：程序设计

适用专业：国际贸易专业（电子商务方向）

一、计算题（20 分）

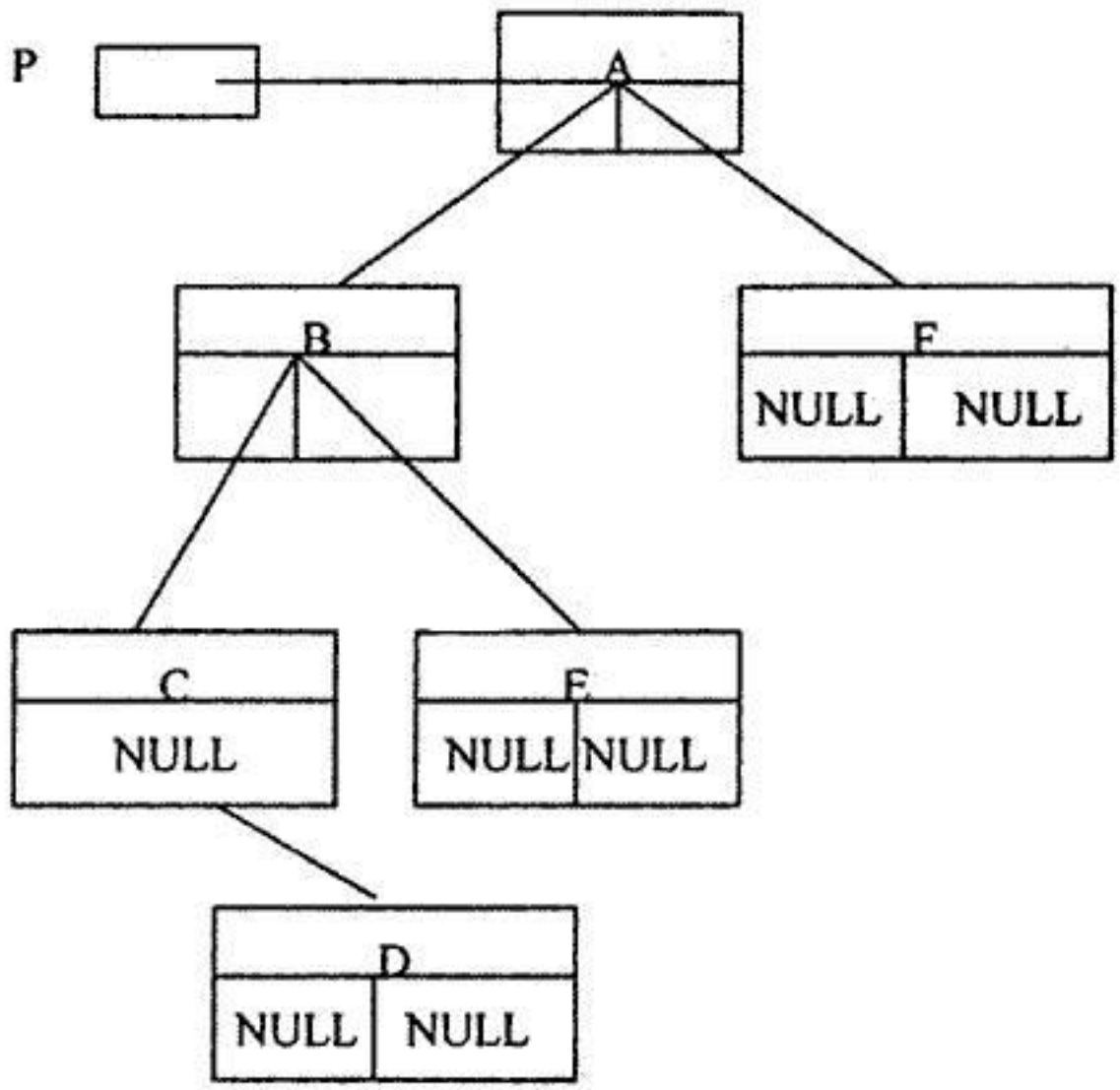
阅读以下程序或程序段，在答题纸上写上程序运行时的输出，

1、当运行如下程序时的输出是_____A_____。

```
Main( )
{int p;
 for (p=10;p<20;p++)
  if(isp(p)&&isp(p+6)&&isp(p+12))
   printf("\n%3d,%3d,%3d",p,p+6,p+12);
  }
  isp(n)
  int n;
  {int i,b=1;
  for (I=2'I<n;i++)
   if(n%I==0)b=0;
  return(b);
  }
```


2、设有二叉树

```
  #define NULL0;
  typedef struct node
  {int c;
  struct node left,right;
  }NODE;
  void preorder(NODE *p)
  {if(p)
   {printf("%c",p->c);
  preorder(p->left);
  preorder(p->right);
  }
 }
void postorder (NODE *p)
 {if (p)
  {postorder(p->left);
  postorder (p->right);
```



```
    printf("%c",p->c);
    }
   }
voial inovder (NODE *P)
{if(p)
 {inovoler (P→left);
  printf("%c",p→c);
  inorder (p→c);
  }
}
```

当执行函数 preorder (NODE*p)时，输出_____B_____，

当执行函数 postorder(NODE*p)时，输出_____C_____，

当执行函数 inorder(NODE*p)时，输出_____D_____。

二、编程题（80 分）

阅读以下程度，将应填入程序中_____XX_____的语句或语句成分写在答卷的对应栏内。

1、该程序是找出[10，100]内能被自己数码积整除的各个数。例如 11、12、15、24、36。（注意，除数不能为零。）

```
 main( )
 {int n;
  printf("\nn=");
  for(n=10;n<100;n++)
   if(_____1_____)
   if(n%prd(n)==0)
    printf("%3d,",n);
 }
 prd(m)
int m;
{int____2_____;
while (____3_____) t_____4_____,m/=10;
return(_____5____);
}
```


2、该程序是在 N*N 的矩阵中安排 1-N*N 各个数。它按矩阵的行列顺时针由外向内安排。例如 n=5 时，安排成：

1	2	3	4	5
16	17	18	19	6
15	24	25	20	7
14	23	22	21	8
13	12	11	10	9

```
main( )
{int a[20] [20];
int n, i, j, k=1;
scanf("%d", &n);
for(i=0;___6____;i++)
 {for (j=i;____7_____;j++)a[i][j]=k++;
  for(____8____;___9____;j++)a[j][____10____]=k++;
  for(____11____;___12____;j--)a[___13____][j]=k++;
  for(____14_____;___15____;j--)a[j][i]=k++;
  }
  if(n%2)a[n/2][n/2]=k;
  for(i=0;i<n;i++)
   {printf("\n");
   for (j=0;j<n;j++)printf("&3d", a[i][j]);
    }
  }
```

3、该程序是遍历 n 个城市车资最少的匈牙利算法。若用矩阵 A 表示车资表，则矩阵元素 aij 表示从 i 城市到 j 城市的车资。该算法通过矩阵的变换和分析来找到遍历的路径。

函数 p 是对分析的矩阵计算“遍历”至少需要的车资。以矩阵的行或列为单位进行减法，保证矩阵每行和每列至少有一个零，这些减数对第 i 行来说是从第 I 城市出发的最少车资；对第 j 列来说是到达第 j 城市的最少车资。例如，

原来矩阵（车资）						进行减法						得到初始矩阵					
*	24	34	14	15	第 1 行各数减	14	*	10	20	0	1
19	*	20	9	6	第 2 行各数减	6	13	*	14	3	0
7	9	*	6	8	第 3 行各数减	6	1	3	*	0	2
34	10	22	*	7	第 4 行各数减	7	27	3	15	*	0
20	8	11	20	*	第 5 行各数减	8	12	0	3	12	*

再第 1 列各数减 1，第 3 列各数减 3。得到初始矩阵（每行每列有 0）：

*	10	17	0	1
12	*	11	3	0
0	3	*	0	2
26	3	12	*	0
11	0	0	12	*

其中，矩阵中*不参加减法，因为*表示自己或已经遍历过的城市。各个减数之和成为这次分析所需要的最小车资。

算法设从“0”号城市出发，将分析矩阵用 B 表示。在 B 初始矩阵基础上，考虑从第 i 号城市到其他各城市的最少车资。若从 i→j，应该矩阵的第 i 行和第 j 列都表示为*，它说明不可能再从第 i 城市出发和由其他城市到达第 j 城市。注消 i 行和 j 列后，对新矩阵

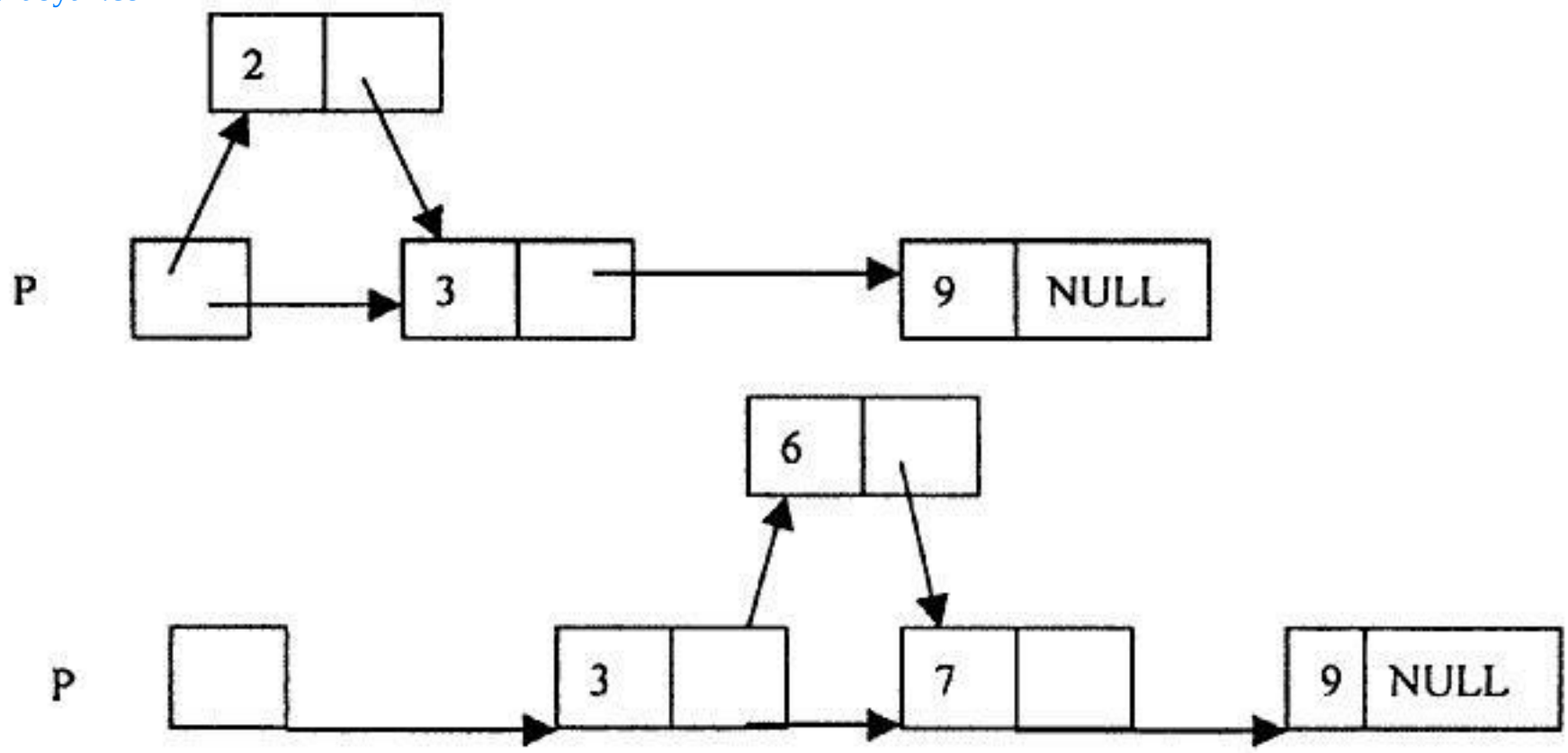
进行分析。从中找到从 i 城市出发最省费用的路径。其中矩阵 C 用于分析，矩阵 D 用于记录最小车资的情况。

算法中数组 path 记录遍历的路径，s 表示遍历过的城市（以*表示）。

```
#define n 5
#define star 1000
typedef int array [n][n];
array a,b,c,d;
int path[n],s[n];
int p(array b,int *m)
{int pi,pj,pk;
____16____;
for(pi=0;pi<n;pi++)/* 对各行进行减法*/
{pk=star;
for(pj=0;pj<n;pj++)
 if(b[pi][pj]<pk)pk=b[pi][pj];
 if((pk>0&&(pk!=star))
{_____17_____;
 for (pj=0;pj<n;pj++)
  if(b[pi][pj]!=star)b[pi][pj]_____18_____;
 }
 }
for(pj=0;pj<n;pj++) /*对各列进行减法  */
{pk=star;
for(pi=0;pi<n;pi++)
 if(b[pi][pj],pk)pk=b[pi][pj];
if((pk>0)&&(pk!=star))
{_____19_____;
 for(pi=0;pi<n;pi++)
 if(b[pi][pj]!=star)b[pi][pj]____20____;
 }
}
return;
}
main(   )
{int I,j,k,min,m1,m,jj,kk,si,sj;
for(I=0;i<n;i++)
 for(j=0;j<n;j++)scanf("%d",&a[i][j]);
for(i=0;i,n;i++)a[i][i]=star;
for(j=1;j,n;j++)s[j]=0;
```

```
 k=0, path[0]=0, I=0, s[0]=star;/* 表示从 0 号城市出发  */
for(si=0;si, n<n;si++)
 for(sj=0;sj<n;sj++)b[si][sj]=a[si][sj];
_____21_____;  /* 分析工作从矩阵 B 开始  */
do{                    /* 分析从第 i 城市到其他可能城市  */
 m1=star;
for(j=0;j<n;j++)
if((s[j]==0)&&(b[i][j]!=star))
{for(si=0;si<n;si++)
  fro(sj=0;sj<n;sj++)c[si][sj]=b[si][sj];
for(kk=0;kk<n;kk++)          /*注消 i 行 j 列       */
 {_____22____;_____23_____;}
p(c,&m);
if(m+b[i][j]<m1)
{____24_____, jj=_____25_____;
 for(si=0;si<n;si++)
  for(sj=0;sj<n;sj++)d[si][sj]=c[si][sj];
 }
}
for(si=0,si<n;si++)  /*接受 ij 城市为达到地，即下次分析的出发地  */
 for(sj=0;sj<n;sj++)b[si][sj]=d[si][sj];
min=____26_____, ____27____, path[++k]=_____28_____;
_____29_____=star;sj=star;
for (si=0;si<n;si++)   /*分析所有城市已经遍历完毕       */
 if(s[si]!=____30____)sj=0;
}while(sj!=star);
printf("\nThe route path is:");
for(i=0;i<n;i++)printf("%d3->",path[i]);
printf("0");
printf("\nTotal of traveling expenses:");
printf("%5d",min);
}
```

4、该程序是读入一级整数（以 0 结束），生成一条由小到大排列的有序链。每读一个非 0 数时，增加一个新结点，并且插入到原来链的相应位置成为新链。例如，（插入考虑原链为空、插入在链首、插入在链尾、插入在链中间。）




```
#define NULL 0;
typedef struct node
  {int c;
   struct node *next;
  }NODE;
main( )
{int n=1;
 NODE p,p1,*p2,*t;
p=NULL;scanf("%D",&n);
while(n)
{t=_____31_____;
t->c=n;scanf("%d",&n);
if(____32_____){_____33_____;t->next=NULL;}
else
 if(t->c<=p->c){____34____;p=t;}
 else
 {____35____;
  do{
   p1=p2;p2=p2->next;
   if(____36_____)
  {p1->next=t;t->next=NULL:}
  else
   if(____37____)
   {p1->next=t;____38_____;p2=NULL;}
  }while(____39_____);
```

```
}/* end of else  */
} /* end of while n * /
printf ("\nP:");pntp(p);
}
pntp(NODE *t)
{
while (_____40_____)
{printf("%3d",t->c);
t=t->next;
}
}
```