```
 /*
 ** This program implements Lab 3: Letters
 **.
 */

#include <stdio.h>
#include <stdlib.h>
#include <time.h>
#include <string.h>
int main()

 {
 FILE    *fpt1,*fpt2;
 unsigned char *image,*MSF,*temp,*check;
 int threshold=0;
 unsigned char *final;
 char    header[320];
 char    header2[320];
 int   ROWS,COLS,BYTES;
 int   ROWS2,COLS2,BYTES2;
 int   r,c,r2,c2,sum,count;
 int   r3,c3;
 char alphabet;
 int row,col;
 FILE *fpt3;

 /* read image */
 if ((fpt1=fopen("parenthood.ppm","rb")) == NULL)
 {
   printf("Unable to open parenthood.ppm for reading\n");
   exit(0);
 }
 fscanf(fpt1,"%s %d %d %d",header,&COLS,&ROWS,&BYTES);
 if (strcmp(header,"P5") != 0  ||  BYTES != 255)
 {
   printf("Not a greyscale 8-bit PPM image\n");
   exit(0);
 }
 if ((fpt2=fopen("temp.ppm","rb")) == NULL)
 {
   printf("Unable to open parenthood.ppm for reading\n");
   exit(0);
 }
 fscanf(fpt2,"%s %d %d %d",header2,&COLS2,&ROWS2,&BYTES2);
 if (strcmp(header2,"P5") != 0  ||  BYTES2 != 255)
 {
   printf("Not a greyscale 8-bit PPM image\n");
   exit(0);
 }

 image=(unsigned char *)calloc(ROWS*COLS,sizeof(unsigned char));
 header[0]=fgetc(fpt1);  /* read white-space character that separates header */
 fread(image,1,COLS*ROWS,fpt1);
 fclose(fpt1);
```

```
/* allocate memory for final version of image */
MSF=(unsigned char *)calloc(ROWS*COLS,sizeof(unsigned char));
header2[0]=fgetc(fpt2);  /* read white-space character that separates header */
fread(MSF,1,COLS*ROWS,fpt2);
fclose(fpt2);
final=(unsigned char *)calloc(ROWS*COLS,sizeof(unsigned char));
temp=(unsigned char *)calloc(15*9,sizeof(unsigned char));
check=(unsigned char *)calloc(15*9,sizeof(unsigned char));

r3=15/2;
c3=9/2;
ROWS2=15;
COLS2=9;

for(threshold=0;threshold<=255;threshold+=5)
{
  count=0;
  for (r=0; r<ROWS; r++)
  {
    for (c=0; c<COLS; c++)
    {
      if(MSF[r*COLS+c]>threshold)
        final[r*COLS+c]= 255;
      else
        final[r*COLS+c]= 0;
    }
  }
  int gt=0,ob=0;
  int tp=0,fp=0,fn=0,tn=0;
  int m=0;
  int k=0,x,y,ov=0;
  int transitions, neighbours, edgecheck, MARKED=1,branch,end;
  fpt3 = fopen("parenthood_gt.txt" , "r");
  r=fscanf(fpt3,"%c %d %d\n",&alphabet,&col,&row);
  while(m != EOF)
  {
  int MARKED=1;
  //printf("%c\n",alphabet );
    if(alphabet=='e')
    {

      gt=1;
    }
    else
    {
    gt=0;
    }

    ob=0;ov=0;

    for (r=row-r3; r<=row+r3; r++)
    {
      for (c=col-c3; c<=col+c3; c++)
      {
```

```
    if(final[r*COLS+c]==255)
    ob=1;
  }
}
if (ob==1)
{
  for (r=row-r3,x=0; r<=row+r3; r++,x++)
  {
    for (c=col-c3,y=0; c<=col+c3; c++,y++)
    {
      temp[x*COLS2+y]=image[r*COLS+c];
    }
  }
  fpt1=fopen("temp1.ppm","w");
  fprintf(fpt1,"P5 %d %d 255\n",COLS2,ROWS2);
  fwrite(temp,COLS2*ROWS2,1,fpt1);
  fclose(fpt1);
  /*
  for(x=0;x<ROWS2;x++)
  {
    for(y=0;y<COLS2;y++)
    {
      printf("%d ",temp[x*COLS2+y]);
    }
    printf("\n");
  }
  */
  for(x=0;x<ROWS2;x++)
  {
    for(y=0;y<COLS2;y++)
    {
      if(temp[x*COLS2+y]<128)
      {
        temp[x*COLS2+y]=0;
      }
      else
      {
        temp[x*COLS2+y]=255;
      }
    }
  }
  fpt1=fopen("temporbin.ppm","w");
  fprintf(fpt1,"P5 %d %d 255\n",COLS2,ROWS2);
  fwrite(temp,COLS2*ROWS2,1,fpt1);
  fclose(fpt1);
  /*
  printf("\ntemp before thinning\n");
  for(x=0;x<ROWS2;x++)
  {
    for(y=0;y<COLS2;y++)
    {
      printf("%d ",temp[x*COLS2+y]);
    }
    printf("\n");
```

```
}
*/

while(MARKED!=0)
{
  //printf("hi");
  MARKED=0;
  for(x=0;x<ROWS2;x++)
  {
    for(y=0;y<COLS2;y++)
    {
      check[x*COLS2+y]=0;
      if(temp[x*COLS2+y]==0)
      {
        transitions=0;

          if((temp[(x-1)*COLS2+(y-1)])==0 && (temp[(x-1)*COLS2+y])==255)
          {  transitions++;     }

          if((temp[(x-1)*COLS2+y])==0 && (temp[(x-1)*COLS2+(y+1)])==255)
          {  transitions++;     }

          if((temp[(x-1)*COLS2+(y+1)])==0 && (temp[(x)*COLS2+(y+1)])==255)
          {  transitions++;     }

          if((temp[(x)*COLS2+(y+1)])==0 && (temp[(x+1)*COLS2+(y+1)])==255)
          {  transitions++;     }

          if((temp[(x+1)*COLS2+(y+1)])==0 && (temp[(x+1)*COLS2+(y)])==255)
          {  transitions++;     }

          if((temp[(x+1)*COLS2+(y)])==0 && (temp[(x+1)*COLS2+(y-1)])==255)
          {  transitions++;     }

          if((temp[(x+1)*COLS2+(y-1)])==0 && (temp[(x)*COLS2+(y-1)])==255)
          {  transitions++;     }

          if((temp[(x)*COLS2+(y-1)])==0 && (temp[(x-1)*COLS2+(y-1)])==255)
          {  transitions++;     }

        else if(x-1)
        neighbours=0;

        if((temp[(x-1)*COLS2+(y-1)])==0)
        {  neighbours++;     }

        if((temp[(x-1)*COLS2+y])==0)
        {  neighbours++;     }

        if((temp[(x-1)*COLS2+(y+1)])==0)
        {  neighbours++;     }

        if((temp[(x)*COLS2+(y+1)])==0)
        {  neighbours++;     }
      if((temp[(x+1)*COLS2+(y+1)])==0)
      {  neighbours++;     }

      if((temp[(x+1)*COLS2+(y)])==0)
      {  neighbours++;     }

      if((temp[(x+1)*COLS2+(y-1)])==0)
      {  neighbours++;     }

      if((temp[(x)*COLS2+(y-1)])==0)
      {  neighbours++;     }

      edgecheck=0;

      if((temp[(x-1)*COLS2+(y)])==255)
      {  edgecheck=1;     }

      else if((temp[(x)*COLS2+(y+1)])==255)
      {  edgecheck=1;     }

      else if((temp[(x+1)*COLS2+(y)])==255 && (temp[(x)*COLS2+(y-1)])==255)
      {  edgecheck=1;     }

      if(transitions==1 && neighbours<=6 && neighbours>=2 && edgecheck==1)
      {
        check[x*COLS2+y]=1;
        MARKED=1;
      }
    }
  }
}
/*
printf("\ncheck\n");
for(x=0;x<ROWS2;x++)
{
  for(y=0;y<COLS2;y++)
  {
    printf("%d ",check[x*COLS2+y]);
  }
  printf("\n");
}
*/

for(x=0;x<ROWS2;x++)
{
  for(y=0;y<COLS2;y++)
  {
    if(check[x*COLS2+y]==1)
    {
    temp[x*COLS2+y]=255;
    }
  }
}
```

```
  /*printf("\ntemp during thinning\n");
  for(x=0;x<ROWS2;x++)
  {
    for(y=0;y<COLS2;y++)
    {
      printf("%d ",temp[x*COLS2+y]);
    }
    printf("\n");
  }
  */
}

  /*
  printf("\ntemp after thinning\n");
  for(x=0;x<ROWS2;x++)
  {
    for(y=0;y<COLS2;y++)
    {
      printf("%d ",temp[x*COLS2+y]);
    }
  printf("\n");
  }
  */

fpt1=fopen("temporthinned.ppm","w");
fprintf(fpt1,"P5 %d %d 255\n",COLS2,ROWS2);
fwrite(temp,COLS2*ROWS2,1,fpt1);
fclose(fpt1);
branch=0;
end=0;
for(x=0;x<ROWS2;x++)
{
  for(y=0;y<COLS2;y++)
  {

    if(temp[x*COLS2+y]==0)
    {
      check[x*COLS2+y]=0;
      transitions=0;

        if((temp[(x-1)*COLS2+(y-1)])==0 && (temp[(x-1)*COLS2+y])==255)
        {  transitions++;     }

        if((temp[(x-1)*COLS2+y])==0 && (temp[(x-1)*COLS2+(y+1)])==255)
        {  transitions++;     }

        if((temp[(x-1)*COLS2+(y+1)])==0 && (temp[(x)*COLS2+(y+1)])==255)
        {  transitions++;     }

        if((temp[(x)*COLS2+(y+1)])==0 && (temp[(x+1)*COLS2+(y+1)])==255)
        {  transitions++;     }

        if((temp[(x+1)*COLS2+(y+1)])==0 && (temp[(x+1)*COLS2+(y)])==255)
```

```
     {  transitions++;     }

     if((temp[(x+1)*COLS2+(y)])==0 && (temp[(x+1)*COLS2+(y-1)])==255)
     {  transitions++;     }

     if((temp[(x+1)*COLS2+(y-1)])==0 && (temp[(x)*COLS2+(y-1)])==255)
     {  transitions++;     }

     if((temp[(x)*COLS2+(y-1)])==0 && (temp[(x-1)*COLS2+(y-1)])==255)
     {  transitions++;     }

    if (transitions>2)
    {
     check[x*COLS2+y]=1;
     branch++;
    }
    if(transitions==1)
    {
     check[x*COLS2+y]=2;
     end++;
    }
   }
  }
 }
 for(x=0;x<ROWS2;x++)
  {
   for(y=0;y<COLS2;y++)
   {
    if(check[x*COLS2+y]==1)
    {
    temp[x*COLS2+y]=180;
    }
    else if(check[x*COLS2+y]==2)
    {
    temp[x*COLS2+y]=90;
    }
   }
  }
 fpt1=fopen("temporbranched.ppm","w");
 fprintf(fpt1,"P5 %d %d 255\n",COLS2,ROWS2);
 fwrite(temp,COLS2*ROWS2,1,fpt1);
 fclose(fpt1);
 if(branch==1&&end==1)
 {
  ov=1;
 }
 else
 {
  ov=0;
 }
 //printf("br:%d end: %d pred: %d\n",branch , end, ov );

}
```

```
      if(gt==1&&ov==1)
      {tp++;}
      else if(gt==0&&ov==1)
      { fp++;}
      else if(gt==0&&ov==0)
      { tn++;}
      else if(gt==1&&ov==0)
      {fn++;}
      m=fscanf(fpt3,"%c %d %d\n",&alphabet,&col,&row);
    }
    fclose(fpt3);
    float tpr,fpr;
    tpr=tp*1.0/((tp+fn)*1.0);
    fpr=fp*1.0/(fp+tn)*1.0;
    //printf("Threshold: %d tp: %d fp: %d tn: %d fn: %d tpr: %f fpr:
%f\n",threshold,tp,fp,tn,fn,tpr,fpr);
    printf("%d\n",fn);
  }

  /* write out final image to see result */
fpt1=fopen("final.ppm","w");
fprintf(fpt1,"P5 %d %d 255\n",COLS,ROWS);
fwrite(final,COLS*ROWS,1,fpt1);
fclose(fpt1);
}
```