```
#include "srgp.h"
#include <stdio.h>

#define LEEFIRST 1
#define LEESECOND 2

#define NOERROR 0
#define DIVBYZERO 1
#define TOOBIG 2
#define LESSTHANZERO 3

int leebuttonIn,leebuttonOut;

char *calcText[20]={"0","/","%","=",
                    "1","2","3","*",
                    "4","5","6","-",
                    "7","8","9","+",
                    "Q"," ","E","C"};
char leenumstring[]="        0";/*9 spaces since 32-bit long*/
int  operflag=LEEFIRST;
long operand[2];
long result;
char leesign;
int  equalflag=1;
int  error=NOERROR;


void Lee_createButton();
void Lee_createCalculator();
int  Lee_convertToArray(long);
char Lee_buttonInput();
void Lee_clearScreen();
void Lee_printArray();
char Lee_getNumber();
void Lee_equalRoutine();


/*************************************************/
void Lee_createButton(void)
{

   leebuttonOut = SRGP_createCanvas(30,30);
   leebuttonIn = SRGP_createCanvas(30,30);

   SRGP_useCanvas(leebuttonOut);
   SRGP_setColor(2);
   SRGP_fillRectangleCoord(0,0,29,29);
   SRGP_setColor(5);
   SRGP_lineCoord(0,29,29,29);
   SRGP_lineCoord(0,28,0,0);
   SRGP_setColor(6);
   SRGP_lineCoord(1,0,29,0);
   SRGP_lineCoord(29,1,29,28);

   SRGP_useCanvas(leebuttonIn);
   SRGP_setColor(2);
   SRGP_fillRectangleCoord(0,0,29,29);
   SRGP_setColor(6);
   SRGP_lineCoord(0,29,29,29);
   SRGP_lineCoord(0,28,0,0);
   SRGP_setColor(5);
   SRGP_lineCoord(1,0,29,0);
   SRGP_lineCoord(29,1,29,28);

   SRGP_useCanvas(SCREEN_CANVAS);
}
/*************************************************/
void Lee_createCalculator(void)
{
  int loopx,loopy,count;
  int width,height,descent;

  SRGP_setColor(3);
```

```
  SRGP_fillRectangleCoord(0,0,180,300);
  SRGP_setColor(0);
  SRGP_fillRectangleCoord(20,235,160,265);
  Lee_createButton();
  count=0;

  for (loopy=15;loopy<=175;loopy+=40)
    for (loopx=15;loopx<=135;loopx+=40)
       {
          SRGP_copyPixel(leebuttonOut, SRGP_defRectangle(0,0,29,29),
                                       SRGP_defPoint(loopx,loopy));
          SRGP_setColor(0);
          SRGP_inquireTextExtent(calcText[count],&width,
                                 &height,&descent);
          SRGP_text(SRGP_defPoint(loopx+15-(width/2),
                                  loopy+15-(height/2)),
                                  calcText[count]);
          count++;
       }
  Lee_clearScreen();
}

/*****************************************************************/
char Lee_buttonInput(void)
{
  int loopx,loopy,count;
  static locator_measure lm,lmup;

  int place;
  int x,y;
  int width,height,descent;

  SRGP_setInputMode (LOCATOR,EVENT);
  place=100;
  while(1)
  {
   do
   {
      if (SRGP_waitEvent(-1)== LOCATOR)
        SRGP_getLocator (&lm);
   } while(lm.buttonChord[0] != DOWN);

   count=0;
   for (loopy=15;loopy<=175;loopy+=40)
     for (loopx=15;loopx<=135;loopx+=40)
     {
       if ((lm.position.x >= loopx) && (lm.position.x <= (loopx+30)))
          if  ((lm.position.y >= loopy) && (lm.position.y <=(loopy+30)))
          {
             place=count;
             x=loopx;
             y=loopy;
             break;
          }
       count++;
     }
     if (place != 100) break;
  }

   SRGP_setColor(0);
   SRGP_copyPixel(leebuttonIn, SRGP_defRectangle(0,0,29,29),
                               SRGP_defPoint(x,y));
   SRGP_inquireTextExtent(calcText[place],&width,
                               &height,&descent);
   SRGP_text(SRGP_defPoint(x+15-(width/2),
                           y+15-(height/2)),
                           calcText[place]);
   do
   {
      if (SRGP_waitEvent(-1)== LOCATOR)
        SRGP_getLocator (&lmup);
   } while(lmup.buttonChord[0] != UP);
```

```
 SRGP_copyPixel(leebuttonOut, SRGP_defRectangle(0,0,29,29),
                                    SRGP_defPoint(x,y));
 SRGP_text(SRGP_defPoint(x+15-(width/2),
                         y+15-(height/2)),
                         calcText[place]);

return *calcText[place];
}

/*****************************************************/

int Lee_convertToArray(long number)
{
  int loop;
  long remainder;

  for (loop=8;loop>=0;loop--)
  {
      remainder=number % 10;
      number /= 10;
      if ((number==0) && (remainder==0))
       leenumstring[loop]=' ';
       else
          leenumstring[loop]=(remainder+48);
       if (leenumstring[8]==' ')
           leenumstring[8]='0';

   }
   if (number !=0)
    return 1; /*overflow*/
  else return 0;

 }
/*****************************************************/

void Lee_clearScreen(void)
{
  SRGP_setColor(0);
  SRGP_fillRectangleCoord(20,235,160,265);
  SRGP_setColor(1);
  SRGP_text(SRGP_defPoint(80,240),"        0");

}

/*****************************************************/

void Lee_printArray(void)
{
  int count;
  SRGP_setColor(0);
  SRGP_fillRectangleCoord(20,235,160,265);
  SRGP_setColor(1);
  SRGP_text(SRGP_defPoint(80,240),&leenumstring[0]);

}

/*****************************************************/

char Lee_getNumber(void)
{
  char numinput;
  char returnchar='5';
  int loop,overflow;
  if (error==NOERROR)
  {
    for (loop=0;loop<=8;loop++)
    {
      numinput=Lee_buttonInput();
      if ((numinput>='0') && (numinput<='9'))
      {
        if (loop==0)
          operand[operflag]=0;
        operand[operflag]=(operand[operflag]*10)+(numinput-48);
```

```
        overflow=Lee_convertToArray(operand[operflag]);
        Lee_printArray();
      } else {returnchar=numinput;break;}
    }
    operflag=LEESECOND;
    while ((returnchar>='0')&&(returnchar<='9'))
      returnchar=Lee_buttonInput();
 }
 else
 {
   error=NOERROR;
   while ((returnchar != 'C')&&(returnchar!='E'))
     returnchar=Lee_buttonInput();
 }
 return returnchar;
}


/*******************************************************/

void Lee_equalRoutine(void)
{
  int overflow;
  switch (leesign)
  {
   case '+': {
               result=operand[LEEFIRST] + operand[LEESECOND];
               break;
             }
   case '-': {
               result=operand[LEEFIRST] - operand[LEESECOND];
               if (result < 0)
                 error=LESSTHANZERO;
               break;
             }
   case '*': {
               result=operand[LEEFIRST] * operand[LEESECOND];
               break;
             }
   case '/': {
               if (operand[LEESECOND]==0)
                error=DIVBYZERO;
               else result=operand[LEEFIRST] / operand [LEESECOND];
               break;
             }
   case '%': {
               if (operand[LEESECOND]==0)
                error=DIVBYZERO;
               else result=operand[LEEFIRST] % operand [LEESECOND];
               break;
              }
  } /*switch*/

 if (error==NOERROR)
 {
   overflow=Lee_convertToArray(result);
   if (overflow==1)
     error=TOOBIG;
   else Lee_printArray();
 }

 if (error != NOERROR)
 {
   SRGP_setColor(0);
   SRGP_fillRectangleCoord(20,235,160,265);
   SRGP_setColor(1);
   switch (error)
   {
     case DIVBYZERO:
     {
       SRGP_text(SRGP_defPoint(20,240),"Error-Div by 0.");
       break;
     }
```

```
      case TOOBIG:
      {
        SRGP_text(SRGP_defPoint(20,240),"Error-Overflow.");
        break;
      }
      case LESSTHANZERO:
      {
        SRGP_text(SRGP_defPoint(20,240),"Error-Negative.");
        break;
      }
    }
  }
}

/*******************************************************/
main ()
{
char getnumchar;
operand[LEEFIRST]=0;
operand[LEESECOND]=0;

SRGP_begin("Calculator",180,300,3,FALSE);


SRGP_loadCommonColor(0, "white");
SRGP_loadCommonColor(1, "black");
SRGP_loadCommonColor(2, "salmon");
SRGP_loadCommonColor(3, "darksalmon");
SRGP_loadCommonColor(4, "sandybrown");
SRGP_loadCommonColor(5, "lightsalmon");
SRGP_loadCommonColor(6, "maroon");

SRGP_setLineWidth(1);

Lee_createCalculator();
leesign='+';

do
{

   getnumchar=Lee_getNumber();
   switch (getnumchar)
{
   case '=':{
                       Lee_equalRoutine();
                       operand[LEEFIRST]=result;
                        operflag=LEEFIRST;
                        equalflag=1;
                        break;
                    }
   case 'C':{
                        leesign='+';
                        operflag=LEEFIRST;
                        operand[LEEFIRST]=0;
                        operand[LEESECOND]=0;
                        Lee_clearScreen();
                        equalflag=0;
                        break;
                     }
  case 'E':{
                       operand[operflag]=0;
                       Lee_clearScreen();
                       equalflag=0;
                       break;
                    }
 case '+':
case '-':
case '*':
case '/':
case'%':{
                     if (equalflag==0)
                     {
                       Lee_equalRoutine();
```

```
                    operand[LEEFIRST]=result;
                  } else equalflag=0;
                  leesign=getnumchar;
                  break;
              }
    }

 } while (getnumchar != 'Q');
SRGP_end();
}
```