* Official Name: Jillian Handrahan
*
* E-mail: jehandra@syr.edu
*
* Assignment: Final Assignment
*
* Environment/Compiler: xcode 14.2
*
* **Date** submitted: April 30, 2023

**thanks to 3Dexport.com for foliage.bmp!*

Reference sheet:

I used gimp to change foliage.png to a 24 bit .bmp file. I loaded the textures into the program using the same code from loadTextures.cpp. This file is cited below in the "UNModified code" section.

Below is examples of my learned code, my modified code with listed sources, and my unmodified code with listed sources, directly from my program.

Learned code:

```
//used loadTextures.cpp to learn how to do this
void loadExternalTextures()
{
  BitMapFile *image[3];
  image[0] = getBMPData("Textures/grass.bmp");
  image[1] = getBMPData("Textures/sky.bmp");
  image[2] = getBMPData("Textures/foliage.bmp");

  // Bind grass image to texture index[0].
  glBindTexture(GL_TEXTURE_2D, texture[0]);
  glTexParameteri(GL_TEXTURE_2D, GL_TEXTURE_WRAP_S, GL_REPEAT);
  glTexParameteri(GL_TEXTURE_2D, GL_TEXTURE_WRAP_T, GL_REPEAT);
  glTexParameteri(GL_TEXTURE_2D,GL_TEXTURE_MIN_FILTER,GL_LINEAR);
  glTexParameteri(GL_TEXTURE_2D,GL_TEXTURE_MAG_FILTER,GL_LINEAR);
  glTexImage2D(GL_TEXTURE_2D, 0, GL_RGB, image[0]->sizeX, image[0]->sizeY, 0,
          GL_RGB, GL_UNSIGNED_BYTE, image[0]->data);
  // Bind sky image to texture index[1]
  glBindTexture(GL_TEXTURE_2D, texture[1]);
  glTexParameteri(GL_TEXTURE_2D, GL_TEXTURE_WRAP_S, GL_CLAMP);
  glTexParameteri(GL_TEXTURE_2D, GL_TEXTURE_WRAP_T, GL_CLAMP);
  glTexParameteri(GL_TEXTURE_2D, GL_TEXTURE_MIN_FILTER, GL_LINEAR);
  glTexParameteri(GL_TEXTURE_2D, GL_TEXTURE_MAG_FILTER, GL_LINEAR);
  glTexImage2D(GL_TEXTURE_2D, 0, GL_RGB, image[1]->sizeX, image[1]->sizeY, 0,
          GL_RGB, GL_UNSIGNED_BYTE, image[1]->data);

   // Bind leaf image to texture index[2]
   glBindTexture(GL_TEXTURE_2D, texture[2]);
   glTexParameteri(GL_TEXTURE_2D, GL_TEXTURE_WRAP_S, GL_REPEAT);
   glTexParameteri(GL_TEXTURE_2D, GL_TEXTURE_WRAP_T, GL_REPEAT);
   glTexParameteri(GL_TEXTURE_2D, GL_TEXTURE_MIN_FILTER, GL_NEAREST);
```

```
    glTexParameteri(GL_TEXTURE_2D, GL_TEXTURE_MAG_FILTER, GL_NEAREST);
    glTexImage2D(GL_TEXTURE_2D, 0, GL_RGB, image[2]->sizeX, image[2]->sizeY, 0,
                 GL_RGB, GL_UNSIGNED_BYTE, image[2]->data);
}
```

(Learned to call the map functions below for bezier curves)

```
 void drawFlower(void) {

    //learned to do this from bezierCurvesWithOutput.cpp
    glMap1f(GL_MAP1_VERTEX_3, 0.0, 1.0, 3, 6, controlPoints[0]);
    glEnable(GL_MAP1_VERTEX_3);
    glColor3f(1.0, 0.0, 1.0);
    glMapGrid1f(100, 0.0, 1.0);
    glLineWidth(5);
    glEvalMesh1(GL_LINE, 0, 100);

    glPushMatrix();
    glTranslatef(40, -45, 0);
    glRotatef(-90, 0, 0, 1);
    glEvalMesh1(GL_LINE, 0, 100);
    glPopMatrix();

    glPushMatrix();
    glTranslatef(-50, -35, 0);
    glRotatef(90, 0, 0, 1);
    glEvalMesh1(GL_LINE, 0, 100);
    glPopMatrix();

    glPushMatrix();
    glTranslatef(-10, -85, 0);
    glRotatef(180, 0, 0, 1);
    glEvalMesh1(GL_LINE, 0, 100);
    glPopMatrix();

    glPushMatrix();
    glColor3f(.3, .2, .1);
    glTranslatef(-5, -40, 0);
    glutSolidSphere(20, 30, 30);
    glPopMatrix();
```

If you used no sources or tools, you must still submit this Reference sheet and affirm that you used no outside sources or tools.

Modified code with files listed above each block:

```
// Control points.
//bezierCurvesWithOutput.cpp MODIFIED to use my points
static float controlPoints[6][3] =
{
    { -16.0, -32.0, 0.0}, { -30.0, 0.0, 0.0}, { -10.0, 7.0, 0.0},
    {0.0, 7.0, 0.0}, {27.0, 0.0, 0.0}, { 7.0, -32.0, 0.0}
};
//particle2.cpp MODIFIED to remove color
typedef struct particle
{
     float position[3];
     float velocity[3];
     float mass;

} particle;

//for loop from particle2.cpp - mass changed
void particleSetUp()
{
    for(int i=0; i<num_particles; i++)
    {
         particles[i].mass = 5.0;

        for(int j=0; j<3; j++)
        {
           particles[i].position[j] = 2.0*((float) rand()/RAND_MAX)-1.0;
           particles[i].velocity[j] = speed*2.0*((float) rand()/RAND_MAX)-1.0;
        }
    }
}
//from particle2.cpp, changed the bounding box and math to suit my bees
void collision(int n)
{
     for (int i=0; i<3; i++)
     {
          if(particles[n].position[i]>=3.0)
          {
               particles[n].velocity[i] = -coef*particles[n].velocity[i];
               particles[n].position[i] = 3-coef*(particles[n].position[i]-3);
          }
          if(particles[n].position[i]<=-3.0)
          {
               particles[n].velocity[i] = -coef*particles[n].velocity[i];
               particles[n].position[i] = -3-coef*(particles[n].position[i]+3);
          }
     }
}
//from myIdle() particle2.cpp - changed to remove repulsion/forces
void beeSystem()
{
    float dt;
    present_time = glutGet(GLUT_ELAPSED_TIME);
    dt = 0.001*(present_time -  last_time);
    for(int i=0; i<num_particles; i++)
    {
       for(int j=0; j<3; j++)
       {
            particles[i].position[j]+=dt*particles[i].velocity[j];
            particles[i].velocity[j]+=dt/particles[i].mass;
        }
        collision(i);
    }
    last_time = present_time;
    glutPostRedisplay();
}


//from color picking example in class MODIFIED heavily to suit my needs and situations
void getID(int x, int y)
{
    unsigned char pixel[3];
    glReadPixels(x, y, 1, 1, GL_RGB, GL_UNSIGNED_BYTE, pixel);

    if ((int)pixel[0]==0&&(int)pixel[1]==255&&(int)pixel[2]==0)
    {
        itemID=BEE;  //green, bee
        if (isSelectBee){
            isInfo = 1;
            isSelectBee= 0;
        }
        else {
            isInfo = 0;
            isSelectBee=1;
        }
        if (isSelectFly) isSelectFly = 0;
        if (isSelectButter) isSelectButter = 0;
    }
    else if ((int)pixel[0]==0&&(int)pixel[1]==0&&(int)pixel[2]==255)
    {
        if (isSelectButter){
            isInfo = 1;
            isSelectButter= 0;
        }
        else {
            isInfo = 0;
            isSelectButter=1;
        }
        if (isSelectFly) isSelectFly = 0;
        if (isSelectBee) isSelectBee = 0;

    }
    else if ((int)pixel[0]==255&&(int)pixel[1]==0&&(int)pixel[2]==0)
    {
        itemID=FLY;  //red, fly
        if (isSelectFly){
            isInfo = 1;
            isSelectFly= 0;
        }
        else {
            isInfo = 0;
            isSelectFly=1;
        }
        if (isSelectBee) isSelectBee = 0;
        if (isSelectButter) isSelectButter = 0;
    }
    else itemID=0;

    selecting=false;
}
```

UNModified code with files listed above each block:

```
//from loadTextures.cpp UNMODIFIED
// Struct of bitmap file.
struct BitMapFile
{
   int sizeX;
   int sizeY;
   unsigned char *data;
};
//from loadTextures.cpp UNMODIFIED
// Routine to read a bitmap file.
// Works only for uncompressed bmp files of 24-bit color.
BitMapFile *getBMPData(string filename)
{
   BitMapFile *bmp = new BitMapFile;
   unsigned int size, offset, headerSize;

   // Read input file name.
   ifstream infile(filename.c_str(), ios::binary);

   // Get the starting point of the image data.
   infile.seekg(10);
   infile.read((char *) &offset, 4);

   // Get the header size of the bitmap.
   infile.read((char *) &headerSize,4);
   // Get width and height values in the bitmap header.
   infile.seekg(18);
   infile.read( (char *) &bmp->sizeX, 4);
   infile.read( (char *) &bmp->sizeY, 4);
   // Allocate buffer for the image.
   size = bmp->sizeX * bmp->sizeY * 24;
   bmp->data = new unsigned char[size];
   // Read bitmap data.
   infile.seekg(offset);
   infile.read((char *) bmp->data , size);

   // Reverse color from bgr to rgb.
   int temp;
   for (int i = 0; i < size; i += 3)
   {
      temp = bmp->data[i];
      bmp->data[i] = bmp->data[i+2];
      bmp->data[i+2] = temp;
   }
   return bmp;
}

//from rendering text example from class UNMODIFIED
void writeBitmapString(void *font, const char *string)
{  const char *c;
   for (c = string; *c != '\0'; c++)
   glutBitmapCharacter(font, *c);
}
```