Computer Systems (SS 2017)
Exercise 3: May 9, 2017

Wolfgang Schreiner
Research Institute for Symbolic Computation (RISC)
Wolfgang.Schreiner@risc.jku.at

January 25, 2017

The exercise is to be submitted by the denoted deadline via the submission interface of the Moodle course as a single file in zip (`.zip`) or tarred gzip (`.tgz`) format which contains the following files:

- A PDF file `Exercise`*`Number`*`-`*`MatNr`*`.pdf` (where *Number* is the number of the exercise and *MatNr* is your "Matrikelnummer") which consists of the following parts:
  1. A decent cover page with the title of the course, the number of the exercise, and the author of the solution (identified by name, Matrikelnummer and email address).
  2. For every source file, a listing in a *fixed width font*, e.g. `Courier`, (such that indentations are appropriately preserved) and an appropriate *font size* such that source code lines do not break.
  3. A description of all tests performed (copies of program inputs and program outputs) explicitly highlighting, if some test produces an unexpected result.
  4. Any additional explanation you would like to give. In particular, if your solution has unwanted problems or bugs, please document these explicitly (you will get more credit for such solutions).
- Each source file of your solution (no object files or executables).

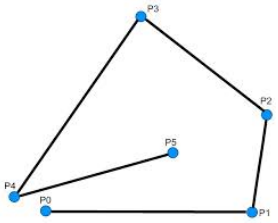
Please obey the coding style recommendations posted on the course site.

Exercise 3: Polygon Courses

A polygon course is a sequence of points connected by straight lines.

1. Write a class `Polygon` that implements polygon courses with the following public interface:

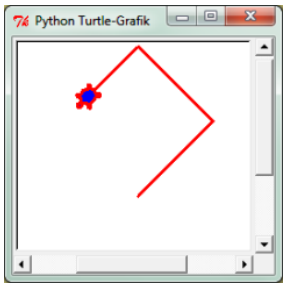
```
class Polygon
{
public:
  // create and destroy polygon
  Polygon();
  ~Polygon();

  // add point with coordinates (x,y) to polygon
  // line from last point to this one has color c (default black)
  void add(double x, double y, unsigned int c = 0);

  // draws the polygon at absolute coordinates (x0,y0) scaled by factor f;
  // thus every point (x,y) is drawn at position (x0+x*f, y0+y*f)
  void draw(double x0, double y0, double f);
};
```



   The class internally maintains a linked list that holds the points of the polygon (objects of some user-defined class `Point`) together with the colors of the lines leading to the points; for drawing the polygon, this list is to be traversed only *once*.

2. Derive from `Polygon` a class

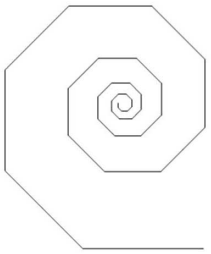
```
class PenPolygon: public Polygon
{
public:
  PenPolygon(double x = 0, double y = 0,
    double a = 0, double c = 0);
  void move(double r);
  void turn(double a);
  void color(unsigned int c);
}
```



   The constructor creates an empty polygon; the polygon maintains a "pen" with color c which is positioned at point (x, y) and is oriented in direction a (angle in radian, 0 is right, $\pi/2$ is up). By the operation `move()` the pen is moved distance r into the current direction, adding a line in the pen color to the polygon; by `turn()`, the pen changes orientation by adding angle a to the current angle; by `color()` the color of the pen is changed to c.

   The class shall make use of the data representation of `Polygon`, i.e. the class must compute the coordinates of the individual points of the polygon and call `add()` to add them to the polygon.

3. Derive from `PenPolygon` a class



```
class SpiralPolygon: public PenPolygon
{
public:
  SpiralPolygon(int n, int s, int c,
    double x, double y, double r, double a,
    double r0, double a0);
}
```

   whose constructor creates a spiral-shaped polygon with n lines; the spiral starts at position (x, y) with a line of length r in direction a; the length of every subsequent line is r_0 times the length of the previous line, the angle of every subsequent line is a_0 plus the angle of the previous line. For instance, with $n = 8$, $r_0 = 1$, and $a_0 = \pi/4$ we get a octagon, chosing $n = 80$, $r0 = 1.1$ and $a_0 = \pi/4$ gives an outward spiral that rotates ten times with eight lines per rotation. The polygon is is drawn in a random color as generated by a random number generator which is initialized with seed s; this color is changed randomly every c lines.

4. Finally write a class

```
class Picture
{
public:
  Picture();
  void add(Polygon *p);
  void draw(double x, double y, double w, double h, double f = 1.0);
}
```

   A `Picture` object represents a rectangular picture which consists of a set of polygons implemented by a linked list of (pointers to) `Polygon` objects. The set is initially empty; the function `add()` adds polygon p to the list. The function `draw()` draws the picture with a rectangular bound with left upper corner (x, y), width w and height h; all polygons are drawn shifted by the position x, y and scaled with factor f.

Write a program that tests these classes by creating a picture, populating it with spirals and drawing the picture in several locations and sizes.