Objects Lab

In this assignment, you are going to modify your old beach hut program to make it more object-oriented.

Part 1

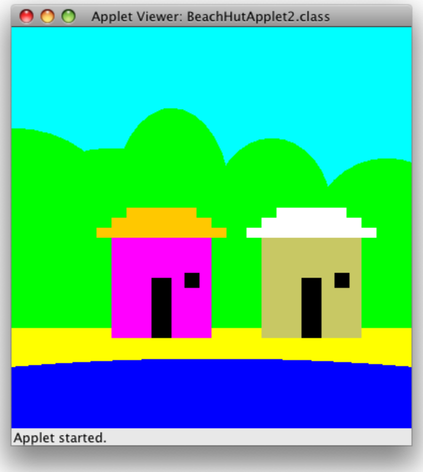
- Create a new class, called `BeachHut` or something similar, that does NOT extend JPanel and does NOT have a `main` method.
- Your `BeachHut` class should have the following fields, constructors, and methods:
  - **FIELDS:**
  - `private Point pos` This variable controls where the beach hut is drawn. (If you prefer, you may use the `Vertex` class we built together, instead of `Point`.)
  - `private Color roofColor` This variable controls what color the beach hut's roof will be.
  - `private Color wallColor` This variable controls what color the hut's exterior wall will be.
  - **CONSTRUCTORS:**
  - `public BeachHut()` This is the default constructor. It assigns the beach hut to a default position and sets the walls and roof to default colors of your choosing.
  - `public BeachHut(int x, int y)` This constructor sets the position to (x,y) as passed in on the parameter list. Roof and walls are set to default colors.
  - **METHODS:**
  - `public void setRoofColor(Color c)` Sets the roof color to the color passed in as `c`.
  - `public void setWallColor(Color c)` Sets the wall color to the color passed in as `c`.
  - `public void draw(Graphics g)` Calls whatever instructions are necessary to draw the beach hut.
- Modify your original "main" (i.e. JPanel-based) class:
  1. Add a new field of type `BeachHut` to your class.
  2. In your constructor, instantiate your `BeachHut` object by calling the zero-parameter "default" constructor.

3. In your `paintComponent` method, keep all the instructions for drawing the background, but delete all the instructions for drawing the beach hut. Instead, call the `draw` method on your `BeachHut` object.

- **STOP.** Run your program, and make sure it works. It should *look* like your first beach hut program, even though it is different internally. Get up; take a walk; get a drink of water. Then begin on Part 2.

Part 2

- Add a second `BeachHut` field to your class. Use the two-parameter (int,int) version of the constructor. Pass it a position that places it somewhere close to the first hut. Give its roof and text different colors than the first beach hut.
- Show the T.A. your code for creating the second beach hut. It should be very short, just calling its constructor and the `setWallColor` or `setRoofColor` methods.
- Here is an example screenshot from the program.



Applet started.

After you have finished:

- Print up the cover page.
- Upload each of your .java files to Canvas.
- Demonstrate your program to a T.A. Please refer to the door of GCB 111 for their office hours.
- Sign your cover page, and turn it in to the TA.

RUBRIC:

1. Show the TA that your program consists of two classes, one that extends JPanel and one that does not. (2 points)
2. Show the TA all the fields in your BeachHut class, and explain what they're used for. (2 points)
3. Show the TA all the constructors in your BeachHut class, and explain when you might want to use one versus another. (2 points)
4. Show the TA all the methods in your BeachHut class, and explain what each one does. (2 points)
5. `BeachHut`'s fields are `private` and its methods and constructors are `public`. (2 points)
6. In your main class, show the TA where you declare both BeachHut objects. (2 points)
7. In your main class, show the TA where you instantiate both BeachHut objects. (2 points)
8. Show the TA that you're using a different constructor for each BeachHut object. (3 points)
9. Show the TA how you customize the color scheme of each BeachHut object. (2 points)
10. Show the TA how you customize the position of each BeachHut object. (2 points)
11. Your program correctly displays two beach huts on the screen. (4 points)
12. Your "main" class uses a `BeachHut` object to draw the hut, instead of doing the drawing itself. (4 points)
13. Each beach hut has a different color scheme. (3 points)
14. Each beach hut has a different position on the screen. (3 points)
15. Each major section of the program is commented. (2 points)
16. Source code is neatly and consistently indented. (2 points)
17. Show the TA that you submitted your code to Canvas. (1 point)

TOTAL: 40 points