Overview: In this lab, you will work in a new (GitHub) group of three (there will be two groups of two), writing C++ code to experiment with inheritance in C++. You will start with a parent Shape class, and then each of you will individually write one of three classes that inherit from the Shape class: Circle, Triangle, and Rectangle.

Groups: You have been assigned at random into a new group (different from last Github lab) as shown below, with your associated GitHub repository listed. The groups with only two members will have Vadim as the third member.

<table>
<thead>
<tr>
<th>team</th>
<th>netids*</th>
<th>repository URL</th>
</tr>
</thead>
<tbody>
<tr>
<td>red</td>
<td>zr9, ts5, ph6</td>
<td><a href="https://github.com/dszajda/cmsc240_s2021_lab7_red">https://github.com/dszajda/cmsc240_s2021_lab7_red</a></td>
</tr>
<tr>
<td>orange</td>
<td>jt9, cg2, tk8</td>
<td><a href="https://github.com/dszajda/cmsc240_s2021_lab7_orange">https://github.com/dszajda/cmsc240_s2021_lab7_orange</a></td>
</tr>
<tr>
<td>yellow</td>
<td>mp5, dt6, rl9</td>
<td><a href="https://github.com/dszajda/cmsc240_s2021_lab7_yellow">https://github.com/dszajda/cmsc240_s2021_lab7_yellow</a></td>
</tr>
<tr>
<td>green</td>
<td>cc7, jp3, ds2</td>
<td><a href="https://github.com/dszajda/cmsc240_s2021_lab7_green">https://github.com/dszajda/cmsc240_s2021_lab7_green</a></td>
</tr>
<tr>
<td>blue</td>
<td>es5, js4, jc7</td>
<td><a href="https://github.com/dszajda/cmsc240_s2021_lab7_blue">https://github.com/dszajda/cmsc240_s2021_lab7_blue</a></td>
</tr>
<tr>
<td>indigo</td>
<td>wb7, ja3</td>
<td><a href="https://github.com/dszajda/cmsc240_s2021_lab7_indigo">https://github.com/dszajda/cmsc240_s2021_lab7_indigo</a></td>
</tr>
<tr>
<td>violet</td>
<td>wn4, rf3</td>
<td><a href="https://github.com/dszajda/cmsc240_s2021_lab7_violet">https://github.com/dszajda/cmsc240_s2021_lab7_violet</a></td>
</tr>
</tbody>
</table>

Getting Started:

- Begin by remotely logging in to one of the mathcs* or turing2 machines, and then create a new lab7 directory within your own directory structure for this course. Then change into your lab7 directory.
- Begin by cloning the original (empty) version of your repository, using your corresponding URL given in the table above:
  
git clone https://github.com/dszajda/cmsc240_s2021_lab7_{YOUR TEAM} 
- Change into the resulting directory.

Group Work:

1. Each person should create, and then switch to, their own non-main working branch.
2. Now create new files according to the following:
   - Person #1: create Shape.h (see course web page) and a Makefile sufficient to compile the group’s work (see the previous GitHub lab’s Makefile for a template)
   - Person #2: create Shape.cpp, implementing the methods outlined in Shape.h
   - Person #3: create tester.cpp, with sufficient tests to demonstrate correctness of Shape.{h,cpp}
3. To stage your work on git:
   - Person #1: git add Shape.h Makefile
   - Person #2: git add Shape.cpp
   - Person #3: git add tester.cpp

4. Each person should then commit their work: git commit -a -m "appropriate message"

5. Checkout the local main branch, merge your working branch, and then push/pull as appropriate from/to the origin main, handling any merge conflicts that git cannot.

6. Again for the three people in your group (each working on their own non-main branch):
   - Person #1: should implement Circle.{h,cpp}, inheriting from Shape. In implementing the print function, include the circle’s (inherited) name, and its radius.
     - Constructor: Circle(std::string name, double radius)
     - For computing area: const long double PI = 3.141592653589793238L;
   - Person #2: should implement Triangle.{h,cpp}, inheriting from Shape. In implementing the print function, include the triangle’s (inherited) name, and its base and height.
     - Constructor: Triangle(std::string name, double base, double height)
   - Person #3: should implement Rectangle.{h,cpp}, inheriting from Shape. In implementing the print function, include the rectangle’s (inherited) name, and its width and height.
     - Constructor: Rectangle(std::string name, double width, double height)
   - Each person should modify the Makefile to accommodate their new class.
   - Each person should add to tester.cpp to test their new class.

7. Once you have fully implemented and tested your class, stage and then commit your work.

8. Checkout the local main branch, merge your working branch, and then push/pull as appropriate from/to the origin main, handling any merge conflicts that git cannot.

---

**Individual Work:** At this point, you should each have a local directory containing working (and identical) code for Shape, Circle, Triangle, Rectangle, and tester.cpp. Now complete the following work on your own individually.

**You should do no more pushes or pulls from the origin main after this point.**

1. Add code similar to the following to tester.cpp (presuming circle, triangle, and rectangle correspond to instances of Circle, Triangle, and Rectangle respectively in tester.cpp):

   ```
   Shape* shapePtr = &circle;
   shapePtr->print();
   shapePtr = &triangle;
   shapePtr->print();
   shapePtr = &rectangle;
   shapePtr->print();
   ```

   Compile, execute, and note the result of the output. In a new README.txt file (inside your team directory), comment on the resulting output — what is being printed for each shape, and why.

2. Now in Shape.h, uncomment the following line:

   ```
   virtual void print();
   ```

   and comment out the previous declaration of print. Compile, execute, and note the result of the output, particularly comparing to the results from the previous step. In your README.txt, comment on the resulting output — what is being printed for each shape, and why.
3. Now in Shape.h, uncomment the following line:

    virtual void print() = 0;

    and comment out the previous declaration of print. (This makes Shape an abstract class.) Compile
    and note the result of the output. In your README.txt, comment on the result — what happens, and
    why.

4. Now change Shape.h back to have:

    virtual void print();

    Compile and execute to ensure that the code runs correctly.

**Submitting:** Package your lab7 directory into a gzipped tarball a la cmsc240_lab7_neitd.tgz and submit
as usual to the appropriate Box folder. The the email address for this lab is

Lab7.mwibsxr9tgepcyim@u.box.com.

**Your lab is due by 5:00 on Thursday, April 1.**