Creative Software Design, Assignment 6-1

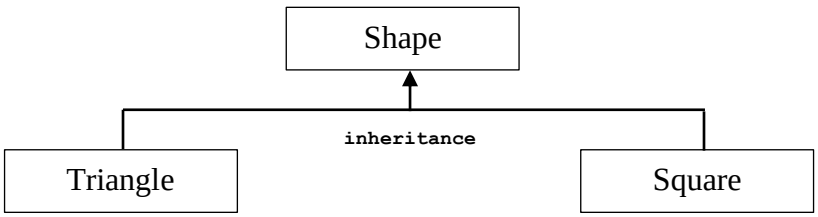
Deadline: 2024-10-15 23:59 (No score for late submission)

- Submit your homework by uploading your zip file to the LMS assignment section. Below is an example.

```
13178_Assignment1-1_2024123456.zip
├─ 1.cc
├─ 2.cc
├─ 3.cc
└─ ...
```

- Your zip file name should follow this format:
  13178_Assignment[Assignment-number]_[Student-ID].zip
  - Ex. 13178_Assignment1-1_2024123456.zip
- Source files should be named as **<filename>.cc** *or* **<filename>.cpp**
- **You must submit your solution in the zip file before the deadline.**

1. **Write a C++ program that find triangle and square areas.**



A. Requirements:

1. Define a base class named `Shape` that has a constructor accepting `width` and `height` values. Both `width` and `height` are of type `int`.
2. Define two subclasses, `Triangle` and `Square`, that inherit the `width` and `height` properties from the `Shape` class.
3. In the `main()` function, create two objects: one of type `Triangle` and one of type `Square`.
4. Both subclasses (`Triangle` and `Square`) should implement a member function called `getArea()`, which returns a `double`.
   i. For the Triangle, the area is calculated as: $(height \times width) \div 2$.
   ii. For the Square, the area is calculated as: $height \times width$.
5. In the `main()` function, use the `getArea()` method to display the areas of both the triangle and the square.

B. Example output of your program (Bold text indicates user input):

```
Width: 2⏎
Height: 5⏎
Area of the Triangle is 5
Area of the Square is 10
```

C. Submission file: one C++ source file (File name: 1.**cc** or 1.**cpp**)

2. Write a C++ program that represents color points in the Cartesian coordinate system `(x, y)`.

A. Define a `Point` class in C++ with the following description:

1. Protected members:
   i. `int x, y`: Represents the x and y coordinates.
2. Public members:
   i. A **constructor** that accepts the x and y coordinate values as arguments (both of type `int`).
   ii. A `move(int _x, int _y)` function that changes the values of `x` and `y` coordinates.

B. Define a `ColorPoint` class in C++ with the following description:

1. Private member:
   i. `string color`: A string representing the color to be assigned to the coordinates.
2. Public members:
   i. The `ColorPoint` class inherits the `x` and `y` coordinates from the `Point` class.
   ii. A `setPoint(int _x, int _y)` function that changes the coordinates by calling the `move` function of the `Point` class.
   iii. A `setColor(string _color)` function that assigns the input color to the `color` attribute.
   iv. A `show()` function that displays the coordinates and the color information.

C. Write a `ColorPoint` class that inherits from the `Point` class so that the following `main()` function runs correctly:

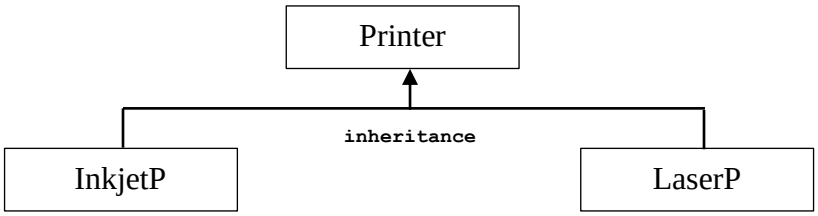
```
int main() {
      ColorPoint cp(5, 5, "RED");
      cp.show();
      cp.setPoint(10, 20);
      cp.setColor("BLUE");
      cp.show();
}
```

D. Example output of your program (Bold text indicates user input):

```
The point's color is RED which is on (5,5).
The point's color is BLUE which is on (10,20).
```

E. Submission file: one C++ source file (File name: 2.**cc** or 2.**cpp**)

3. Write a C++ program that designs a class with an inheritance structure for different types of printers, as shown in the diagram.



A. Define a `Printer` class in C++ with the following description:

1. Private members:
   i. `int availableCount`: Represents the ink or toner capacity of the printer.
   ii. `int availablePage`: Represents the number of papers available in the printer.
2. Public members:
   i. A **constructor** that accepts `availableCount` and `availablePage` as arguments (both of type `int`).
   ii. A `print(int usedPage)` function that updates the remaining `availableCount` and `availablePage` values after printing.
      - One unit of toner or ink is used per page
   iii. `getAvailableCount()`: Returns the `availableCount` value.
   iv. `getAvailablePage()`: Returns the `availablePage` value.

B. Define an `InkjetP` class in C++ with the following description:

1. Private members:
   i. `string model, manufacturer`: A constant string for the model and manufacturer information.
2. Public members:
   i. The class inherits the `availableCount` and `availablePage` values from the `Printer` class.
   ii. A **constructor** that initializes the printer's information (`model`, `manufacturer`, `availableCount`, and `availablePage`.)
   iii. A `print(int usedPage)` function that checks whether printing is possible using the base `Printer` class's print function:

- If printing is successful, display “Printed.”
- If printing fails due to insufficient paper, display “Cannot print because of not enough page.”
- If printing fails due to insufficient ink, display “Cannot print because of not enough ink.”

iv. A `showInfo()` function that displays the printer’s information (`model`, `manufacturer`, `availableCount`, and `availablePage`).

C. Define a `LaserP` class in C++ with the following description:

1. Private members:

   i. `string model, manufacturer`: A constant string for the model and manufacturer information.

2. Public members:

   i. The class inherits the `availableCount` and `availablePage` values from the `Printer` class.

   ii. A **constructor** that initializes the printer’s information (`model`, `manufacturer`, `availableCount`, and `availablePage`).

   iii. A `print(int usedPage)` function that checks whether printing is possible using the base `Printer` class’s print function:

   - If printing is successful, display “Printed.”
   - If printing fails due to insufficient paper, display “Cannot print because of not enough page.”
   - If printing fails due to insufficient toner, display “Cannot print because of not enough toner.”

   iv. A `showInfo()` function that displays the printer’s information (`model`, `manufacturer`, `availableCount`, and `availablePage`).

D. **Define a `main()` function:**

1. Initially, each printer sets the number of pages and ink/toner.
2. Allow the user to select a printer and input the number of pages to print.
3. After each print, ask “Do you want to keep printing? (y/n)” and display the `InkJetP` and `LaserP` information.
4. If the user selects ‘y’, accept another input for the number of pages to print and the ink/toner to use.

5. If the user selects ‘n’, terminate the program.

E. Example output of your program (Bold text indicates user input):

```
Currently operating 2 printers are follow
5 10⏎
InkJet: Officejet V30, HP, 5 available pages, 10 available Ink
3 20⏎
Laser: SCX-6x47, Samsung, 3 available pages, 20 available Toner

Insert printer(1:InkJet, 2:Laser) and pages: 1 4⏎
Printed.
InkJet: Officejet V30, HP, 1 available pages, 6 available Ink
Laser: SCX-6x47, Samsung, 3 available pages, 20 available Toner
Do you want keep printing?(y/n)>> y⏎

Insert printer(1:InkJet, 2:Laser) and pages: 2 10⏎
Cannot print because of not enough pages.
InkJet: Officejet V30, HP, 1 available pages, 6 available Ink
Laser: SCX-6x47, Samsung, 3 available pages, 20 available Toner
Do you want keep printing?(y/n)>> y⏎

Insert printer(1:InkJet, 2:Laser) and pages: 2 2⏎
Printed.
InkJet: Officejet V30, HP, 1 available pages, 6 available Ink
Laser: SCX-6x47, Samsung, 1 available pages, 18 available Toner
Do you want keep printing?(y/n)>> n⏎
```

F. Submission file: one C++ source file (File name: 3.**cc** or 3.**cpp**)