Object-Oriented Programming Homework #2   **Jan 27th, 2023**

Working with C++ Functions

1. Write a program that read words from standard input and transform those words before printing to output.

1.1) Write the function `unstylize` which is used to remove an enclosing "*…*" pairs from a word and complete the program that read words from input and print "unstylize" words to output.

1.2) Modify the program from 1.1) to print words in **reverse order** to output.

1.3) Modify the program from 1.1) to print words in **alphabetical order** to output.

Input	Output for 1.1)	Output for 1.2)	Output for 1.3)
C *C++* Rust* *Python* * *Java	C C++ Rust* Python * *Java	*Java * Python Rust* C++ C	* *Java C C++ Python Rust*

2. Given the following example score sheet that records scores for every play for each player:

```
Leo 5000 1200 2000 4500
Mike 3800 2400 3200
Raph 1500 2200 1200 4000 4800
Don 5000 1600 3200 4600
May 4400 3300 5800
```

2.1) Write a program that reads and store the player's name with all of his/her scores, then generate the rankings based on **the maximum score** played by each player. The player with higher score has higher rank. On equal rank, order the display by the player name.

2.2) Write additional programs for generating the rankings based on **(1) the minimum score** and **(2) the average score** played by each player. The player with higher score has higher rank. On equal rank, also order the display by the player name.

2.3) Partition the program from 2.1) to 2.2) to have **at least** one header file for all utility functions used by the program, one source file for definitions of all utility functions, and one source file for each program. **Create CMakeLists.txt** file for the project, configure and build all programs specified in the project. Finally, test the programs by running all of them.

Rankings table generated from programs should be as shown below:

Max Score Rankings	Min Score Rankings	Average Score Rankings
1. May 5800 2. Don 5000 3. Leo 5000 4. Raph 4800 5. Mike 3800	1. May 3300 2. Mike 2400 3. Don 1600 4. Leo 1200 5. Raph 1200	1. May 4500 2. Don 3600 3. Leo 3175 4. Mike 3133.33 5. Raph 2740

3. Given the following SVG image file as a template:

```
<svg width="500" height="500" xmlns="http://www.w3.org/2000/svg">
  <rect width="100%" height="100%" fill="#EEEEEE" />
  <circle cx="250" cy="250" r="250"
          stroke="black" stroke-width="2" fill="none" />
  <circle cx="250" cy="250" r="10" fill="#00FFFF" />
</svg>
```

3.1) Write a program that takes the number N from user input and use it to generate N points $p_i = (x_i, y_i)$ where p_i lies within the unit circle.

3.2) Partition the program from 3.1) to have **at least** one header file for all utility functions used by the program, one source file for definitions of all utility functions, and one source file for the program. **Create CMakeLists.txt** file for the project, configure and build the program. Finally, test the program.

Hint: Use polar coordinate system to ensure that each random point lies within the unit circle.

Advice: Use I/O redirection to save the output to a file for viewing from the browser.