VG101 — Introduction to Computer and Programming

Homework 8

Manuel — UM-JI (Fall 2020)

- MATLAB: write each exercise in a different file
- C/C++: use the provided assignment template
- Include simple comments in the code
- If applicable, split the code over several functions
- Extensively test your code and impove it
- Write a single README file per homework
- Archive the files (`*.{zip|tar}`) and upload on Canvas

*Exercises preceded by a * are mandatory. Any student not completing all of them, or submitting a work that cannot compile or be interpreted will automatically be deducted 1 mark on the final course grade.*

JOJ Online Judge

Exercises 1 to 1 can be tested on JOJ Online Judge.

Important reminders regarding the Online Judge (OJ):

- For each exercise save all the files, without any folder structure, into a `.tar` archive;
- Strictly stick to the input and output formats provided in the specifications;
- The OJ only checks the correctness of the code not its quality;
- For feedbacks on the quality, submit the code as part of the assignment and include the OJ score as well as the failed cases in the README file;

* **Ex. 1** — *Stack, vector, queue, and array*

1. Write three C++ functions which read strings from the standard input and print them in the reverse order. The first program should be implemented using an array, the second one using a vector and the third one using a stack. Argue on the best choice in the README file.
2. Write three C++ functions which read strings from the standard input and print them in the input order. The first program should be implemented using an array, the second one using a vector and the third one using a queue. Argue on the best choice in the README file.

Specifications.

- Input: a single line containing space separated strings
- Output: a single line containing space separated strings
- JOJ extra rules:
  - Add `#include "assignment.h"`
  - Do not include the `main()` function
  - The function names should be `ex1_reverse_array`, `ex1_reverse_vector`, `ex1_reverse_stack`, `ex1_ordered_array`, `ex1_ordered_vector`, and `ex1_ordered_queue`

* **Ex. 2** — *Class implementation*

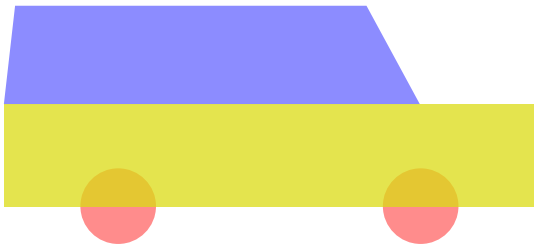
1. Following the definition of the `Circle` class in chapter 10, write and implement the following classes: `Triangle`, `Rectangle`, `Parallelogram`, and `Trapezium`.

2. Reorganise and adjust the above classes to take advantage of inheritance and polymorphism.
3. Add the necessary code to effectively draw the above shapes using OpenGL.

* **Ex. 3 —** *Classes and OpenGL*

Write a C++ class to draw and move a car using the geometrical classes from exercise 3. The car should be similar to the one below, and should move back and forth from left to right. This can be easily achieved by moving all the geometrical object it is composed of. More complex car shapes can be implemented.

Hint: the display can be updated using the function `glutTimerFunc`



Ex. 4 — *Course survey*

Complete the course survey as instructed by the undergraduate office and get a +2 bonus on your lab grade.