Assignment 1 Part 3 – A Templated Collection Class

Objectives: *This assignment gives you some experience with designing and writing C++ using the generic programming approach.*

- (5 points) Create a file, called README (Template provided as LYX file. You are free to use any software but stick to the format given in the templated file):
  - Submit to eCampus an electronic version of the file README by **September 24th**.
  - Test the C++ programs on CSE Linux machine.
  - The assignment will be graded focusing on: program design and correctness.
  - You will be given a .zip file containing these files:
    `Collection.h`, `collection_test.cpp`, `makefile`,
    `Stress_ball.cpp`, `Stress_ball.h`, `Stress_ball_test.cpp`, `Stress_ball_test.h`,
    `Stress_ball1.data`, `Stress_ball2.data`,
    `Jeans.cpp`, `Jeans.h`, `Jeans_test.cpp`, `Jeans_test.h`, `Jeans1.data`, `Jeans2.data`
  - Use `collection_test.cpp`, `Jeans_test.cpp`, `Stress_ball_test.cpp` to test your implementation.
  - When your program works correctly, **upload only `Collection.h`, `Jeans.cpp`, `Stress_ball.cpp`, `Jeans_test.cpp`, `Stress_ball_test.cpp`**, and **`makefile`** to Mimir Classroom by September 24th where your program will be tested against TA's test cases.
    - **Do not upload any other files.**
    - **Do not use the main() function in any of the given files except `collection_test.cpp` file .**

Problem Description (95 pts)

1. (45 points) Write a templated version of the class `Collection` with the template parameters: `Obj`, `F1`, `F2`.

   (a) The templated class `Collection` and all the templated functions should be in the header file `collection.h` (there is no `collection.cpp` file in this assignment). You have to transfer **all** the function definitions from `collection.cpp` (from previous part) to `collection.h` (except the input `operator>>)` .

   (b) Replace the class `Stress_ball` as (typename) `Obj`, `Stress_ball_colors` as `F1`, and `Stress_ball_sizes` as `F2`. You are given a file, `Stress_ball_test.cpp`. Complete this file by filling dots (...) such that it can be used with the templated `Collection` class. In order not to use long class names, use aliases:

   ```
   using CollectionSB = Collection<Stress_ball, Stress_ball_colors,
                                   Stress_ball_sizes>;
   ```

   (c) The input `operator>>` can be templated but you need to use a specific version for each template class. So for the class `Stress_ball` use this approach:

   ```
   istream& operator>>(istream& is, CollectionSB& c);
   ```

   where you explicitly use the class `Stress_ball` (do not use the template parameters `Obj`, `F1`, or `F2`). And do not put it in the file `Collection.h` but put it in the file `Stress_ball_test.cpp`.

   (d) Use the same `Stress_ball` class created in the previous parts of the assignment.

2. (30 points) Write a class `Jeans` identical to `Stress_ball` class. You are given a header file `Jeans.h`. Implement these methods in `Jeans.cpp` exactly identical to `Stress_ball.cpp.`

   (a) Apply the `Collection` functions to `Jeans` objects using `jeans_test.cpp`.

   (b) Complete the given test file `jeans_test.cpp` by filling dots (...) such that it can be used with the templated `Collection` class. In order not to use long class names, use aliases:

   ```
   using CollectionJN = Collection<Jeans, Jeans_colors, Jeans_sizes>;
   ```

   (c) For the input `operator>>`, use the class `Jeans` explicitly (do not use the template parameters `Obj`, `F1`, or `F2`):

   ```
   istream& operator>>(istream& is, CollectionJN& c);
   ```

   Do not put it in the file `collection.h` but use it in the file `jeans_test.cpp`.

   (d) You may use `collection_test.cpp` for testing your code against the files you have completed.

3. (5 points) You are given a skeleton `makefile`. "`test`" is the name of an executable file that is used in `makefile`. Please do not change this. Complete `makefile` and upload it to mimir.

4. (15 points) Based on this assignment Part 3 write about the generic programming using templates in your README file.

The C++ program must be submitted to Mimir Classroom.
The README file must be submitted to eCampus by September 24th.
You should test all the implemented functions/operators.