LINKÖPING UNIVERSITY
Department of Computer and Information Science
Software and Systems
Eric Elfving 2016-03-31

Computer examination in **TDDD38** Advanced Programming in C++

Date 2016-03-31

Time 14-19

Department IDA

Course code TDDD38

Exam code DAT1

Examiner

Klas Arvidsson (klas.arvidsson@liu.se)

Administrator

Anna Grabska Eklund, 28 2362

Teacher on call

Eric Elfving (eric.elfving@liu.se, 013-28 2419)
Will primarialy answer exam questions using the student client.
Will only visit the exam rooms for system-related problems.

Allowed Aids (tillåtna hjälpmedel)

An English-* dictionary may be brought to the exam.
No other printed or electronic material are allowed.
The cplusplus.com reference is available in the exam system.

Grading

The exam has a total of 25 points.
0-10 for grade U/FX
11-14 for grade 3/C
15-18 for grade 4/B
19-25 for grade 5/A.

Special instructions

- Don't log out at any time during the exam, only when you have finished.
- Given files are found in subdirectory `given_files` (write protected).
  The exam will be available as a pdf in this directory at the start of the exam.
- Files for examination must be submitted via the Student Client, see separate instructions (`given_files/student_client.pdf`)!
- When using standard library components, such as algorithms and containers, try to chose "best fit" regarding the problem to solve. Avoid unrelated/unnecessary computations and unnecessary data structures.
- C style coding may cause point reduction where C++ alternatives are available.
- Make sure that your code compiles. Commented out regions of non-compiling code may still give some points.

Theory questions

Answers may be given in either Swedish or English. Submit your answers to all theory questions in one text file called `THEORY.TXT` and submit it as `Assignment #1`.

1. Given the following declarations, explain why the call `max(i,d)` will give an error and give a possible fix in the call to make it compile. [1p]

   ```
   template <typename T>
   T max(T, T);

   int i;
   double d;
   ```

2. In C++, it is possible to create an unnamed namespace. What is an unnamed namespace and what is it used for? [1p]
3. Give two different uses of the keyword `mutable` and describe the effect of `mutable` in each case. [1p]
4. Give two ways of supplying the compiler with implicit type conversion to or from a user-defined type. [1p]
5. Give an example of a delegating constructor. [1p]

Programming exercises

6. Submit your solution as `Assignment #6` [5p]

The game Yahtzee is based on throwing five dice and trying to get specific combinations of these dice. In this assignment, you are going to implement a part of the scoring mechanism.

There are a lot of possible combinations that gives points in Yahtzee, but in this exercise we are going to support checking for a number of ones, number of twos and one pair (two dice with the same result) in a given set of dice.

Create an abstract base class `Score_Variant` having two direct subclasses `Counted_Dice` and `Pair`. Also create two subclasses to `Counted_Dice` called `Ones` and `Twos`.

All classes must have these two member functions:

- string name() - Gives the string "Ones", "Twos" and "Pair" for the classes Ones, Twos and Pair respectively.
- **int** score() - Takes a vector<**int**> and returns the highest possible score for this scoring variant. For Ones and Twos, this is the sum of all ones and twos, for Pair the result is the sum of the highest possible pair in the vector. See table 1 for examples.

Since the calculation in `score` for Ones and Twos will be very similar, this calculation should be done by `Counted_Dice` by adding a constructor taking an int (the die searched for) and storing this as a member. `Counted_Dice` will also have a member function `get_number` to get the value of this number (this may be useful in the final printout step).

No other member functions than those listed are allowed.

It must not be possible to copy or move any objects of the above class types.

Dice	Ones	Twos	Pair
`1, 2, 3, 2, 1`	2	4	4
`3, 3, 5, 5, 5`	0	0	10

Table 1: Expected scores from the three variants for some given sets of dice.

Also create a main function that test your implementation by creating a vector containing one of each of the concrete classes (Ones, Twos and Pair). Also create one vector of ints representing your dice. For each of the score variants, check if the dice gives any points and if so, print first the name of the score variant, then the score according to that variant and, if it is Ones or Twos, print the number of ones or twos found. An example is given in listing 1.

Listing 1: Exmple output for the dice 3, 2, 2, 3, 5

```
Twos: 4 (2)
Pair: 6
```

Note: Even if this assignment only require three concrete score variants to be implemented, your code must show some general solutions so that it can be easily expanded with more score variants. Note that all possible `Conted_Dice` should print the number of scoring dice in the set (not specific to `Ones` and `Twos`).

7. Copy the file `given_files/assignment7.cc` to your working directory and make changes to your copy. Submit your answer as `Assignment #7` [5p]

   There is a function that works a bit like the standard algorithm `std::find` (but much simpler).

```
int * Find(int * start, int * end, int val)
{
    for ( auto it = start; it != end; ++it )
    {
        if ( *it == val )
        {
            return it;
        }
    }
    return end; // wasn't found
}
```

   Your task is to do the following modifications to the given function so that the commented lines in the given code compile and give output according to the comments.

   - Create a namespace called `assignment` and rename the given function to `find` and add it to that namespace to get rid of possible name clashes with `std::find`.
   - Make `assignment::find` into a template that works with real iterators instead of just pointer to int.
   - Change the third argument of `assignment::find` to a unary predicate function (a function taking one argument and returning `bool`). `find` shall return the first element for which the predicate returns `true` (or the second argument if none is found).

   With these alterations, we have gone from a very basic linear find that searches for a specific value of type int to a version of `std::find_if`.

8. Write your code in a file named `assignment8.cc` and submit your answer as `Assignment #8`. [5p]

   In this assignment, you're supposed to show good knowledge of the standard library. Any usage of standard iteration statements will give point deductions. In each step of your solution, make sure to use the algorithm most suited to solve the task – `for_each` is not a valid solution for all problems.

   Here is a model competitive programming exercise for you to solve according to the algorithm presented after the problem text.

```
Input: two numbers N and T followed by N integer values separated by spaces
       (this range is called A), a blank line and T integers.
Output: For each integer t (1<=t<=N) from the T integers, print the sum
        of the t first integers in A.

Example:

Input:
5 3
1 5 2 3 6

3
1
4

Output:
8
1
11
```

   This task shall be solved in the following way. For each step, do nothing more and nothing less.

   1. Read the two integers `N` and `T` using standard formatted input.
   2. Create a vector, called `input_values` to store the range `A`.
   3. Read the N integers and store them in `input_values`.
   4. Create a new vector `sums`.
   5. Calculate the accumulated sums from `input_values` so that index `i` in `sums` contains the sum of all values of `input_values` in the index range $[0, i]$.
   6. For each of the `T` remaining integers (it's okay to read to end of input), print index `t-1` from `sums` to `cout`.

   Note: To get full marks on this exercise, you must make sure to prohibit unnecessary reallocations of the vectors during input and summation.

9. Copy the file `given_files/assignment9.cc` to your working directory and make changes to that file. Submit your answer as `Assignment #9`. [5p]

   This task is divided into two parts, the first giving three of the total five points and the second is worth the two remaining points. You are not supposed to give separate solutions for the two parts, but may stop after implementing the first part.

   Your task in this assignment is to create a stream output manipulator to format a container for printing to some `ostream`. When done, the code in listing 2 shall compile and give output according to Listing 3.

   Listing 2: Code example for assignment 9

```
int main()
{
    vector<int> vec {2, 5, 1, 7, 10};
    cout << format_container() << vec << endl;
    cout << format_container('{') << vec << endl;
}
```

   Listing 3: output from listing 2

```
[2, 5, 1, 7, 10]
{2, 5, 1, 7, 10}
```

   (a) 1. Create a simple struct called `format_container` with three data members; a pointer to an `ostream`, and starting and ending delimiters. Add a constructor taking a char to set the starting delimiter (it's enough to support brackets and braces) with brackets being the default. [3p]
   2. Create an output operator that takes an ostream and a `format_container` that sets the `ostream` pointer for that `format_container` and returns it.
   3. Create another output operator that takes a `format_container` and a `vector` and prints the vector on the given stream according to the example above.

   (b) Generalize your second output operator by making it into a template parameterized on the container type. You can assume that there is an existing overload of the output operator for the element type (the type you get when dereferencing an iterator for the given container). After modifications, the code in listing 4 shall compile. [2p]

   Listing 4: Code after generalization

```
list<string> lst{"hi", "does", "this", "work?"};
forward_list<int> fl{3,65,1,8};
cout << format_container() << lst << "\n"
     << format_container() << fl << endl;
```