OHJ-1156 Programming II

Class Exam
Monday 09.05.2011

Imed Hammouda

Make sure you read the questions carefully before giving your answer. Put your name and student number on each answer sheet.
This exam consists of 3 pages / 7 exercises. The maximum amount of points is 70. Each exercise is worth 10 points.
Written material, mobile phones and calculators are NOT allowed in the exam.
Good luck!

Exercise 1

The ADT Calendar has so far been defined as follows:

```
class Calendar {
  public:
    Calendar();
    ~Calendar();
  private:
    struct Entry {
      string task_name;
      Entry* next;
      Entry* prev;
    };
    Entry* first;
    Entry* find( string& name );
};
```

Add a specification and give an implementation of three member functions:
1. to add a new element to the list if it is not already found.
2. to remove an element. The function should return *false* if the element is not found.
3. to count the amount of elements with a specified name in the list.

Exercise 2

True or false?
1. A pointer variable cannot store the address of another pointer variable.
2. The last element in a string is always the null character.
3. The new operator returns a pointer to a variable and obtains memory for a new variable from the free store.
4. If *p* is a pointer variable, the command *delete p* ends the lifetime of *p*.
5. The only values that can be used to initialize a pointer are *0* and *NULL*.
6. In C++, a function contained within a class is called a member function.
7. Protecting data from access by unauthorized functions is called data abstraction.
8. When each object of a class maintains its own copy of an attribute, the variable that represents the attribute is also known as an instance.
9. If not otherwise specified, all class members are by default public.
10. Creating an actual function from a template is called overloading.

Exercise 3

Explain shortly the following concepts. An example alone is not enough.
1. Dynamic data
2. Node of a linked structure
3. Generic data structure
4. Control flow graph
5. Standard exception