System and Device Programming
Standard Exam
13.01.2025

Ex 1 (1.25 points)

Suppose to run the following program. Report the exact output generated by the program with no extra symbols or characters.

```
std::map<int, int> f (const std::vector<int>& vec) {
      std::map<int, int> m;
      for (const int& elem : vec) {
      ++m[elem];
      }
      return m;
}

int main() {
  std::vector<int> vec = {1, 1, 1, 1, 2, 2, 2, 3, 3, 4, 5, 5};
      std::map<int, int> m = f(vec);
      for (const auto& pair : m) {
      std::cout << pair.first << "[" << pair.second << "]";
      }
      return 0;
}
```

Answer:

Ex 2 (1.25 points)

Suppose to run the following program. Indicate which of the following statements are correct. Note that incorrect answers imply a penalty in the final score.

```
using namespace std;
using std::cout;
using std::endl;

class C {
private:
  …
public:
  ...
};

int main() {
 cout << "{1}"; C e1, e2;
 cout << "{2}"; C *e3 = new (nothrow) C[10];
 cout << "{3}"; C &e4 = e1;
 cout << "{4}"; e2 = e1;
 cout << "{5}"; e3[0] = e1;
 cout << "{6}"; e4 = (std::move(e2));
 cout << "{7}"; delete [] e3;
 cout << "{8}"; return 1;
}
```

Choose one or more options:

1. ⬜ Line number 1 includes two copy constructors.
2. ⬜ Line number 2 includes ten standard constructors.
3. ⬜ Line number 2 includes one standard constructor.
4. ⬜ Line number 4 includes one copy assignment operator.
5. ⬜ Line number 4 includes one move assignment operator.
6. ⬜ Line number 6 includes one move assignment operator.
7. ⬜ Line number 7 includes ten destructors.
8. ⬜Line number 8 includes three destructors.

Ex 3 (1.25 points)

Suppose to run the following program. Report the exact output generated by the program with no extra symbols or characters.

```
#include <iostream>
#include <vector>
#include <string>

int main() {
    int a = 10;
    double b = 20.5;
    std::vector<int> vec = {1, 2, 3, 4, 5};

    auto lambda1 = [a](int x) {
    return a + x;
    };

    auto lambda2 = [&b](double y) {
    b += y;
    return b;
    };

    auto lambda3 = [&vec](int index) -> int {
    if (index >= 0 && index < (int) vec.size()) {
        return vec[index];
    }
    return -1;
    };

    auto lambda4 = [a, b]() {
    return a * b;
    };

    std::cout << lambda1(5) << " ";
    std::cout << lambda2(10.5) << " ";
    std::cout << lambda3(2) << " ";
    std::cout << lambda4() << " ";

    return 0;
}
```

Answer:

Ex 4 (2.0 points)

Explain what a condition variable in C (or C++) is and how it is typically used with mutexes. Describe a typical use case scenario for condition variables in a producer-consumer problem. Explain why each construct must be inserted in the overall scenario.

Ex 5 (2.5 points)

Specify what are templates in C++ and why they are helpful.

Show an example of how they can be applied to create a class template managing a list container. On the list, the class templates should allow inserting new elements, extracting existing components, and displaying the entire list content.

Moreover, specify how definitions and declarations must be divided into the ".c", ".h", and, possibly, ".hpp" files.

Ex 6 (3.25 points)

Implement a program in C++ running four threads:

- The first thread awaits a random variable time (from 1 to 10 seconds) and then generates the first atom of Hydrogen (H).
- The second thread awaits a random variable time (from 1 to 10 seconds) and then generates a second atom of Hydrogen (H).
- The third thread awaits a random variable time (from 1 to 10 seconds) and then generates an atom of Oxygen (O).
- The fourth thread produces a water molecule (H2O) when the two hydrogen and oxygen atoms are available.

To synchronize the four threads, use promises and futures**.**

Ex 7 (3.5 points)

Write a program in C++ that implements the following producer and consumer schemes.

There are two producers, process P1 and process P2. P1 produces elements in the FIFO named FA. P2 produces elements in the FIFO named FB. Suppose that P1 and P2 generate strings (they can read them from standard input or a file) at random intervals (ranging from 1 to 10 seconds).

Consumer C consumes an element from FA, then one from FB, and then restarts the entire cycle from FA. It displays all strings it reads on standard output (or save in a file).

The entire process ends once the consumer receives two consecutive strings equal to "end" (one from FA and one from FB).