System device and programming

Exam 16/01/2023 C++

Exercize 08 (2.5 pt)

Write a C++ program that operates on a vector of integers v.
You should manage the correct synchronization of the following threads:

- a thread “writer” that adds a random number between 1 and 10 to the vector every 5 seconds;
- a thread “worker” that executes the commands received from console (if valid);
- a thread “ui” that constantly checks for user input from console; the valid commands are the following ones:
  - 0 = terminate the program;
  - 1 = print all elements in the vector;
  - 2 = print last element of the vector;
  - 3 = delete all elements from the vector.

The function to put a thread in the sleep status (e.g., for 1 second) is the following one:

```
std::this_thread::sleep_for (std::chrono::seconds(1))
```

Write the code of the program and manage threads synchronization.
Make sure all threads finish running before the main program terminates.

If you do not remember the exact syntax of C++ synchronization primitives, you can write down a mock version (with same sense...). Correctness is strictly required in the template syntax which is required to be right, as well as in any basic C++ syntax.

Exercise 06 (1.0 pt)

What are the elements of vector v after the execution of the while cycle?
Here is the signature of the function memcpy: void *memcpy(void *__dst, const void *__src, size_t __n)
Note that a wrong answer might imply a negative score

- 0,2,4
- 0,1,2
- 1,2,3
- 2,3,4
- 2,4,6

```
#include <vector>
using namespace std;

int main() {
    int i = 0;
    vector<int> v{1,2,3};
    auto l = [&](int& a){ memcpy(&v[i],&a,1*sizeof(int));};

    int temp;
    while( i<3 ){
        temp = v[i]*2;
        cout << temp << endl;
        l(temp);
        i++;
    }
}
```

Exercise 07 (1,5 pt)

In which lines of the main the move constructor is called?
Note that a wrong answer might imply a negative score

- Line 1
- Line 2
- Line 3
- Line 4
- It is never called

```
#include <iostream>
using namespace std;

class Y {

public: //the five copy-control members
    //constructors
    Y()  { std::cout << "dc " << std::endl; } //default constructor dc
    Y(const Y &) {  std::cout << "cc" << std::endl; } //copy constructor cc
    Y(Y &&) noexcept { std::cout << "mc" << std::endl; }; //move constructor mc
    //assignments
    Y &operator=(const Y &) { std::cout << "ca" << std::endl; } //copy
assignment ca
    Y &operator=(Y &&) {std::cout << "ma" << std::endl;} //move assignment ma
    //destructor
    ~Y() { std::cout << "d" << std::endl; } //destructor d
};

Y* f_a(){ return new(Y);}
Y f_b(Y& y_b){ return Y(y_b);}

int main() {

    Y y0;     // line 1

    Y *y1 = f_a(); // line 2

    Y y2 = f_b(y0); // line 3

    delete(y1); // line 4

    return 0; //line 5
}
```

SOLUTION FOR STANDARD VERSION

A) A POSSIBLE SOLUTION.

Check the provided code.

B) ANSWER

Check the provided code.

C) ANSWER

Move constructor is never called because default constructors are used to build y0 and the new object inside f_a, while copy constructors is used to return a copy of y_b inside function f_b. See also the provided code.