System and Device Programming
Standard Exam
07.07.2023

Ex 1 (1.5 points)

Suppose that the following program is run using the command
`./pgrm abc 2`
Indicate the possible output or outputs of the program. Note that wrong answers imply a penalty in the final score.

```
#define L 100

int main (int argc, char *argv[]) {
  int i, j;
  char str1[L], str2[L];
  setbuf(stdout,0);
  i = atoi (argv[2]);
  for (j=0; j<i; j++) {
    if (fork () == 0) {
      sprintf (str1, "echo -n [%d]", j);    // "-n" indicates no "new line"
      system (str1);
    } else {
      printf ("{%d}", j);
      sprintf (str1, "%s", argv[1]);
      sprintf (str2, "%d", j);
      execlp (argv[0], "myPgrm", str1, str2, NULL);
    }
  }
  return (0);
}
```

Choose one or more options:

1. ⬜ `[1]{1}[0]{0}{1}[1]`
2. ⬜ `[0]{0}[0]{0}{1}[1]`
3. ⬜ `{0}[0]{1}{0}[1][0]`
4. ⬜ `{0}[0]{0}{1}[1][0]`
5. ⬜ `[0]{0}[1]{1}{0}[0]`
6. ⬜ `{0}[0][1]{1}[0]{0}`
7. ⬜ `{0}{0}{1}[1][0][0]`

Ex 2 (1.5 points)

Analyze the following code snippet in C++. When the main is executed, indicate how many (standard) constructors, copy constructors, and destructors are called.

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

int main() {
  C e1;
  C e2 = e1;
  C e3 = *new C;
  return 0;
}
```

Choose one or more options:

1. ⬜ 1 constructor, 2 copy constructors, and 3 destructors.
2. ⬜ 1 constructor, 1 copy constructor, and 2 destructors.
3. ⬜ 2 constructors, 2 copy constructors, and 4 destructors.
4. ⬜ 2 constructors, 2 copy constructors, and 3 destructors.
5. ⬜ 3 constructors, 1 copy constructor, and 3 destructors.
6. ⬜ 3 constructors, 2 copy constructors, and 3 destructors.
7. ⬜ 1 constructor, 2 copy constructors, and 2 destructors.

Ex 3 (1.5 points)

Analyze the following code snippet in C++. Indicate the possible output or outputs obtained by executing the program. Note that wrong answers imply a penalty in the final score.

```
auto lambda = []( std::string h )->bool{
   return ( h != "-" && h != "." );
};

int main() {
   std::string s("123.456.789-00");
   std::vector<std::string> num;
   for (int i = 0; i < s.length() ; i++) {
      num.push_back( s.substr(i, 1) );
   }
   cout << s << "#";
   for( auto z : num ){ if (lambda(z)) std::cout << z; }; std::cout << '\n';
   return 0;
}
```

Choose one or more options:

1. ⬜ The program displays the sequence “`123.456.789-00#`“
2. ⬜ The program displays the sequence “`123.456.789-00#..-`“
3. ⬜ The program does not run as there is a bug.
4. ⬜ The program displays the sequence “`123.456.789-00#12345678900`”
5. ⬜ The program displays the sequence “`123.456.789-00#123.456.789-00`”
6. ⬜ The program displays the sequence “`12345678900#12345678900`”

Ex 4 (2.5 points)

A text (ASCII) file stores the information concerning a set of students. For each student, one line of the file indicates the following information: The register number, last and first name (we suppose all students have only two names), the number of examinations passed, and the mark for each one of those exams. The following is a correct example of such a file:

```
100000 Granger Hemione 8 30 30 30 29 29 30 30 29
124567 Potter Harry 5 30 18 24 29 28
113567 Weasley Ron 4 28 26 27 28
…
```

Write a C++ function that:

- Receives the file name as a parameter.
- Stores the file content in a hash map of sets. Each element of the hash table stores: The register number (which is also the key of the hash table), the last and first name (standard fields), and the set of the marks received by the student (into a field of type set).

The function must return a reference to the data structure it has created.

Ex 5 (2.5 points)

Describe how to use C++ templates and which problem they solve. Report an example to manipulate a FIFO list of different types (booleans, integers, floats, strings, etc.). Illustrate how to use this template class (i.e., write the client program) and indicate at least two different ways to organize it into the header (.h) and the source (.cpp) file.

Ex 6 (2.5 points)

The Pthread library implements binary semaphores with the functions `pthread_mutex_init`, `pthread_mutex_lock`, and `pthread_mutex_unlock`. Using these functions, implement a counting semaphore, i.e., a non-binary semaphore, whose starting value is `count`.

Ex 7 (3.0 points)

A C (or C++) program executes four threads: TA, TB, TC, and TD. These threads are cyclical, run forever, and cooperate to generate sets of symbols on subsequent lines of the standard output. Each one of them can display one single character (an `'A'`, `'B'`, `'C'`, or `'D'`, respectively, and eventually a new line) for each iteration of their main cycle. Each line must have the following format:

$A^2\ \{B|C\}^{2+}\ D^2$

This means that for each sequence, there are:

- A^2: Exactly two symbols A.
- $\{B|C\}^{2+}$: Two or more symbols B or C (i.e., at least `BB`, `CC,` `BC`, or `CB`, and then maybe others `B` or `C)`.
- D^2: Exactly two symbols `D`.

Each sequence is terminated by a “new line” character.
The following is a correct example of the execution of such a program:

```
AABCDD
AACBDD
AACBCDD
AABCCDD
AABCBCCCDD
AACBCBDD
...
```