System and Device Programming
Standard Exam
20.06.2023

Ex 1 (1.5 points)
Suppose the following program is run using the command:
```
./pgrm 2
```
Indicate a possible program output.

```c
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <unistd.h>
#include <sys/wait.h>

int main (int argc, char *argv[]) {
  int i, n;
  char str[40];

  n = atoi (argv[1]);

  for (i=1; i<=n; i++) {
    if ( fork() == 0) {
      sprintf (str, "%d", n-1);
      execlp (argv[0], argv[0], str, NULL);
    }
  }

  printf ("%d", n);
  fflush (stdout);

  exit (0);
}
```

Choose one or more options:

1. ☐ 21100
2. ☐ 221100
3. ☐ 01012
4. ☐ 210210
5. ☐ 110022
6. ☐ 11200
7. ☐ 2110

Ex 2 (1.5 points)
Indicate the possible output, or outputs, that can be obtained by concurrently executing the following processes PA, PB, and PC with the reported semaphore initialization.

```
init (S1, 0);
init (S2, 1);
```

PA	*PB*	*PC*
wait(S1);	wait(S2);	wait(S2);

```
printf("A");              printf("D");              printf("E");
signal(S2);               signal(S1);               signal(S1);
wait(S2);
printf("B");
wait(S1);
printf("C");
```

Choose one or more options:

1. ☐ DABC
2. ☐ DABCE
3. ☐ EABC
4. ☐ DAE
5. ☐ EABCD
6. ☐ DAD
7. ☐ EAE
8. ☐ DAEBC
9. ☐ EAD
10. ☐ EADBC

Ex 3 (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.

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

void f1(C e) { ... }
void f2(C &e) { ... }

int main() {
   C e1, e2;
   f1(e1);
   f2(e2);
   C *e3 = new C;
   return 0;
}
```

Choose one or more options:

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

Ex 4 (2.5 points)

Describe lambda expressions and why they have been introduced in C++ (which problem do they solve?). Illustrate the meaning of the capture list, the parameter list, the return type, and the body. Report some examples to illustrate these features.

Ex 5 (3.0 points)

Use barriers implemented with semaphores, mutexes, and counters to implement the following synchronization scheme:
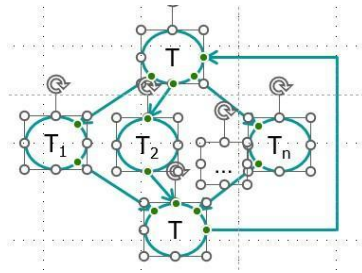
In the previous figure, T and all T_i are different cyclic (i.e., running a primary cycle) threads.

Ex 6 (3.0 points)

Use tasks promises and future to implement three threads T_1, T_2, and T_3 executed directly by the main thread such that:
- T_1 reads a string from standard input, transfers that string to T_2, and terminates.
- T_2 transforms the string received from T_1 into capital letters, transfers it to T_3, and terminates.
- T_3 displays the string received from T_2 on standard output and terminates.

Indicate how it is possible to make the three threads cyclic, i.e., repeat the process reported in the previous itemization (the C code in not necessary, but it is requested a description in English language).

Ex 7 (2.0 points)

Indicates the main differences between FIFO, message queues, and sockets. Describe when it is better to use each one of the methods instead of the others and which are the main steps to implement it (the C code in not necessary, but it is requested a description in English language).