TP 1 - Partie 2 - Exercice 5 - I

```cpp
#include <cassert>
#include <iostream>

#define SHOW(arg) std::cout << "Macro SHOW “" #arg "”: " << (arg) << '\n';

// Returns the fibonacci number using recursive evaluation: the algorithm
// is (very) inefficient as the same number is evaluated many times.
int fibonacci(int n) {
  assert(n >= 0);
  if (n <= 1)
    return 1;
  return fibonacci(n - 1) + fibonacci(n - 2);
}

// Returns the fibonacci number using iterative evaluation.
int fibonacci_iterative_evaluation(int n) {
  assert(n >= 0);
  if (n <= 1)
    return 1;
```

TP 1 - Partie 2 - Exercice 5 - II

```
20      int f_nm1{1};
21      int f_nm2{1};
22      int f;
23      for (int i{2}; i <= n; ++i) {
24        f = f_nm1 + f_nm2;
25        f_nm2 = f_nm1;
26        f_nm1 = f;
27      }
28      return f;
29    }
30
31    int main() {
32      SHOW(fibonacci(0))
33      SHOW(fibonacci_iterative_evaluation(0))
34      SHOW(fibonacci(1))
35      SHOW(fibonacci_iterative_evaluation(1))
36      SHOW(fibonacci(2))
37      SHOW(fibonacci_iterative_evaluation(2))
38      SHOW(fibonacci(3))
39      SHOW(fibonacci_iterative_evaluation(3))
```

TP 1 - Partie 2 - Exercice 5 - III

```
40      SHOW(fibonacci(4))
41      SHOW(fibonacci_iterative_evaluation(4))
42  }
```

Output:

```
1   Macro SHOW “fibonacci(0)”: 1
2   Macro SHOW “fibonacci_iterative_evaluation(0)”: 1
3   Macro SHOW “fibonacci(1)”: 1
4   Macro SHOW “fibonacci_iterative_evaluation(1)”: 1
5   Macro SHOW “fibonacci(2)”: 2
6   Macro SHOW “fibonacci_iterative_evaluation(2)”: 2
7   Macro SHOW “fibonacci(3)”: 3
8   Macro SHOW “fibonacci_iterative_evaluation(3)”: 3
9   Macro SHOW “fibonacci(4)”: 5
10  Macro SHOW “fibonacci_iterative_evaluation(4)”: 5
```