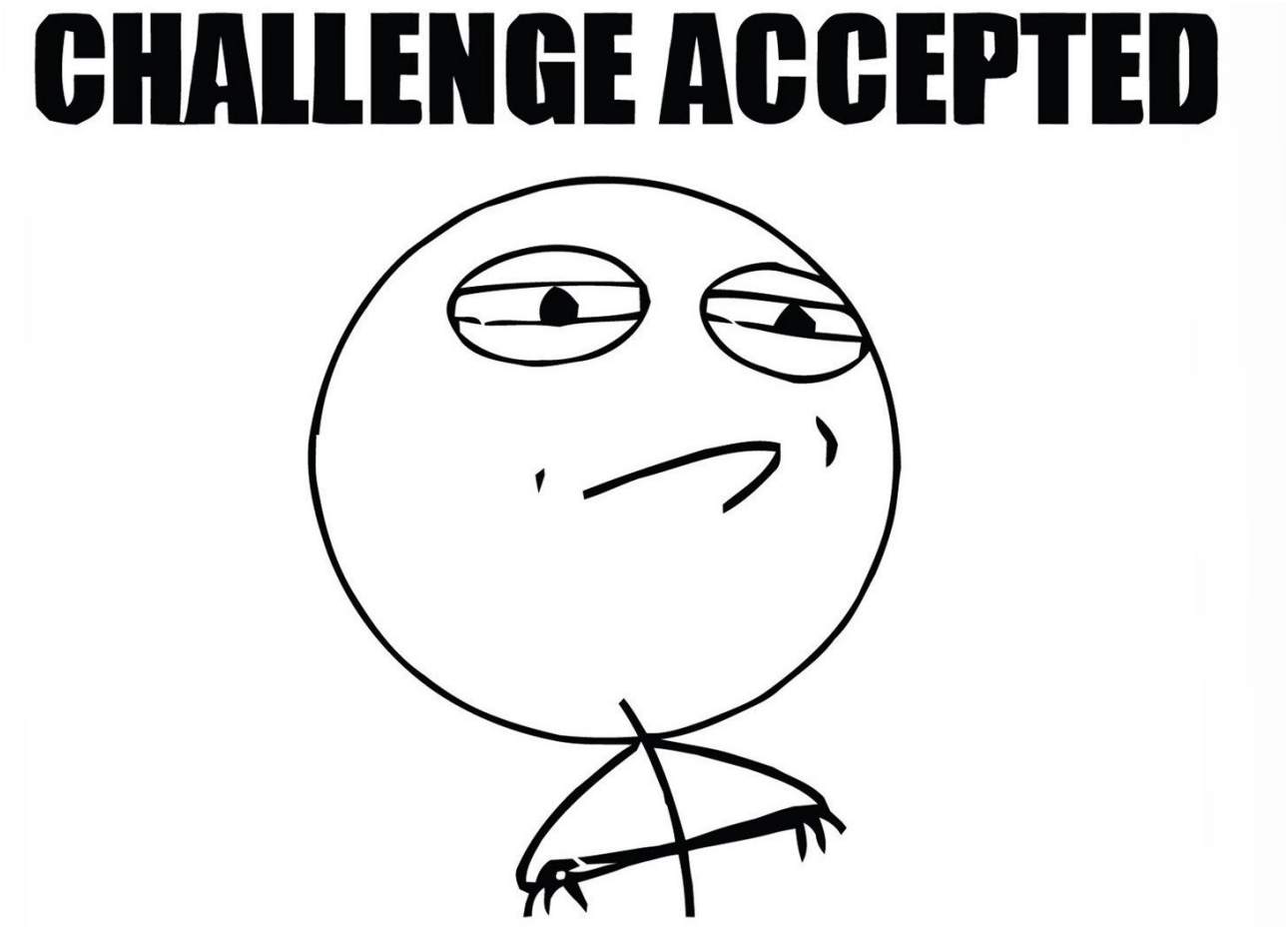
PROGRAMANDO

Aprendiendo a pensar como las máquinas



Manel Guerrero

Índice

1 Iterando

1 Nomenclatura

Precondición: Aquello que se tiene que cumplir antes del bucle (o de cualquier trozo de código) para que el programa funcione correctamente.

Postcondición: Aquello que se cumple al salir del bucle (o después de cualquier trozo de código).

Invariante: Condición que se cumple antes de entrar en un bucle y después de cada iteración.

Función de cota: Función que indica el número máximo de iteraciones que nos quedan por hacer. Si es muy complicada de calcular aceptaremos una cota superior (un número que seguro que será mayor al número máximo de iteraciones).

2 Problemas

- Haz un programa que imprima la suma de 5 enteros que el usuario introducirá por teclado.

- Haz un programa que calcule el factorial del número que el usuario le pida.

Factorial:
- 0! = 1
- 1! = 1
- n! = n * (n-1)!

- Haz un programa que imprima los primeros n elementos de la serie de Fibonacci.

Fibonacci:
- F0 = 0
- F1 = 1
- Fn = Fn-1 + Fn-2

0, 1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89, 144, ...

- Haz un programa que calcule el máximo común divisor por el algoritmo de Euclides.

Algoritmo de Euclides:
- mcd(a, 0) = a
- mcd(a, b) = mcd(b, a%b)

3 Suma de N enteros

```
#include <stdio.h>

#define NUMS     5

main() {
      int i, n, sum;

      printf("Introduce %d enteros:\n", NUMS);
      printf("PRE:\tNUMS = %d\ti = %d\tn = %d\tsum = %d\n",
                  NUMS, 0, n, 0);
      for (i = 0, sum = 0; i < NUMS; i++) {
            scanf("%d", &n);
            sum += n;
            printf("ITER:\tNUMS = %d\ti = %d\tn = %d\tsum = %d\n",
                        NUMS, i, n, sum);
      }
      printf("POST:\tNUMS = %d\ti = %d\tn = %d\tsum = %d\n",
                        NUMS, i, n, sum);
      printf("Suma = %d\n", sum);
}
```

```
Precondición del bucle: NUMS>=0
Inicialización: i = 0 (iteradora), sum = 0 (acumuladora)
Invariante: sum = suma de los números leídos hasta esta iteración.
Función de cota: NUMS - i
Postcondición: sum = suma de todos los números leídos.
```

```
$ ./a.out
Introduce 5 enteros:
PRE:  NUMS = 5    i = 0 n = ? sum = 0
5 7 2 4 1
ITER: NUMS = 5    i = 0 n = 5 sum = 5
ITER: NUMS = 5    i = 1 n = 7 sum = 12
ITER: NUMS = 5    i = 2 n = 2 sum = 14
ITER: NUMS = 5    i = 3 n = 4 sum = 18
ITER: NUMS = 5    i = 4 n = 1 sum = 19
POST: NUMS = 5    i = 5 n = 1 sum = 19
Suma = 19
$
```

4 Factorial

```
#include <stdio.h>

main() {
      int i, n, f;

      printf("Cálculo del factorial de 'n'. Introduce 'n': ");
      scanf("%d", &n);
      printf("PRE:\tn = %d\ti = %d\tf = %d\n", n, 2, 1);
      for (i = 2, f = 1; i <= n; i++) {
            f *= i;
            printf("ITER:\tn = %d\ti = %d\tf = %d\n", n, i, f);
      }
      printf("POST:\tn = %d\ti = %d\tf = %d\n", n, i, f);
      printf("%d! = %d\n", n, f);
}
```

```
Precondición del bucle: n>=0
Inicialización: i = 2 (iteradora), f = 1 (acumuladora)
Invariante: f = i!
Función de cota: n - i - 1
Postcondición: f = n!.
```

```
$ ./a.out
Cálculo del factorial de 'n'. Introduce 'n': 5
PRE:  n = 5 i = 2 f = 1
ITER: n = 5 i = 2 f = 2
ITER: n = 5 i = 3 f = 6
ITER: n = 5 i = 4 f = 24
ITER: n = 5 i = 5 f = 120
POST: n = 5 i = 6 f = 120
5! = 120
```

5 Fibonacci

```
#include <stdio.h>

main() {
      int i, n, f1, f2, f3;

      printf("Cuantos números de la serie de Fibonacci quieres: ");
      scanf("%d", &n);
      if (n >= 1) {
            printf("0");
            if (n >=2) {
                  printf(", 1");
                  printf(" PRE:\ti = %d n = %d f1 = %d f2 = %d f3 = %d\n",
                              0, n, 0, 1, f3);
                  for (i = 3, f1 = 0, f2 = 1; i <= n; i++) {
                        f3 = f1 + f2;
                        f1 = f2;
                        f2 = f3;
                        printf(", %d", f3);
                        printf(" ITER:\ti = %d n = %d f1 = %d f2 = %d f3 = %d\
n",
                                    i, n, f1, f2, f3);
                  }
            }
      }
      printf(" POST:\ti = %d n = %d f1 = %d f2 = %d f3 = %d\n",
                   i, n, f1, f2, f3);
      printf(".\n");
}
```

```
Precondición del bucle: n >= 0
Inicialización: i = 3 (iteradora), f1 = 0, f2 = 1 (antepenúltimo y último
elementos de la serie calculados; f3 será el siguiente a calcular).
Invariante: f3 = fibo(i), f2 = fibo(i-1), f1 = fibo(i-2).
Función de cota: n - i - 1
Postcondición: f3 = fibo(n).
```

Ejemplo de ejecución con los “chivatos”:

```
$ ./a.out
Cuantos números de la serie de Fibonacci quieres: 5
0, 1 PRE:   i = 0 n = 5 f1 = 0 f2 = 1 f3 = -1077072388
, 1 ITER:   i = 3 n = 5 f1 = 1 f2 = 1 f3 = 1
, 2 ITER:   i = 4 n = 5 f1 = 1 f2 = 2 f3 = 2
, 3 ITER:   i = 5 n = 5 f1 = 2 f2 = 3 f3 = 3
    POST:   i = 6 n = 5 f1 = 2 f2 = 3 f3 = 3
.
```

Ejemplo de ejecución sin los “chivatos”:

```
$ ./a.out
Cuantos números de la serie de Fibonacci quieres: 5
0, 1, 1, 2, 3.
$
```

6 Máximo Común Divisor

```
#include <stdio.h>

main() {
      int a, b, old_a;

      printf("MCD(a,b):\n");
      printf("a: ");scanf("%d", &a);
      printf("b: ");scanf("%d", &b);
      while(b) {
            old_a = a;
            a = b;
            b = old_a % b;
            printf("a = %d\tb = %d\n", a, b);
      }
      printf("MCD = %d\n", a);
}
```

```
Precondición del bucle: a>=0 && b>=0
        (No es necesario que a >= b)
Invariante: a == mcd(a,b) * k1 && b == mcd(a,b) * k2
        (siendo k1 y k2 constantes enteras)
Función de cota: b (se pueden obtener funciones más acotadas, como ahora a%b,
        pero es obvio que b disminuye rápidamente)
Postcondición: a == mcd(a,b) && b == 0
```

```
$ ./a.out
MCD(a,b):
a: 654
b: 2322
PRE:  a =       654     b = 2322
ITER: a =      2322     b = 654
ITER: a =       654     b = 360
ITER: a =       360     b = 294
ITER: a =       294     b = 66
ITER: a =        66     b = 30
ITER: a =        30     b = 6
ITER: a =         6     b = 0
POST: a =         6     b = 0
MCD = 6
$
```