Standard de codificación

Departamento de Informática

Universidad Nacional de Rosario, Instituto Politécnico, Dto. de Informática,
informática@ips.edu.ar,
WWW home page: http://informatica.ips.edu.ar

Resumen Mediante el presente documento se busca estandarizar el modo de escribir y presentar un programa. Si bien los ejemplos puestos están en código C, este documento es válido para cualquier lenguaje que se use en cualquier materia dentro del Departamento abarcando ambos niveles, el medio y el superior.
La finalidad de esto es escribir código legible, ordenado y adquirir el hábito de una correcta presentación del trabajo. Su uso será obligatorio y evaluado.
Este documento está basado en el que implemento la cátedra de Programación II de la carrera Licenciatura en Ciencias de la Computación de la Facultad de Ciencias Exactas, Ingeniería y Agrimensura de U.N.R.

1. Construcción de Programas

1.1. Pasos a cumplir

1. Diseño de datos
2. Signatura y declaración de propósito
3. Ejemplos de lo pretendido
4. Definición de funciones
5. Evaluación y testing

Diseño de datos Debe ser una descripción del programa, como se representarán los datos, que funcionalidad y objetivo tiene la aplicación, función o conjunto de ellas de forma general.

Hay que prestar principal atención a responder en la descripción a ¿cómo representaremos la información?

Signatura y declaración de propósito Aquí indicaremos que parámetros recibe (cuantos y de que tipo), y qué datos retorna. La descripción de próposito indica una breve descripción del ¿qué hace? la función.

En el caso de los problemas que abordaremos debemos decidir:

1. cuáles son los datos de entrada que se nos proveen;
2. cuáles son las salidas que debemos producir, y
3. cuál es la relación entre todos ellos.

Se hace una salvedad con respecto a este tema, el cuál también es valido a los otros apartados, Hay tener en cuenta que dependiendo el lenguaje en que se programe, los comentarios y el modo de escritura variarían. Vale este breve ejemplo en funcional:

```
# Representamos temperaturas mediante únmeros enteros
# farCel: Int -> Int
# El áparmetro representa una temperatura en Fahrenheit y,
# se retorna su equivalente en Celsius.
```

Ejemplos de lo pretendido Luego del paso anterior, podemos saber el resultado de la función para algunos valores de entrada. Debe quedar explícito cual es la entrada y cual debería tener que ser el resultado.

Defínción de funciones ¡Escribimos el código!

Traducimos todo lo diseñado a un lenguaje de programación.

Evaluación y Testing Aplicamos el código generado con los ejemplos diseñados previamente, se verifica que los resultados coincidan con lo que se esperaba. Estos resultados deben quedar registrados sean correctos o no.

Corrección o modificaciones en caso de error Si algunas de las pruebas anteriores fallan, se deberá revisar y corregir el código. Luego, volver a testear.

2. Ejemplos

2.1. Fahrenheit a Celsius

Problema Escribir un programa que utilice la fórmula $^{o}C = (5/9)(^{o}F - 32)$ para mostrar grados Fahrenheit a Celsius.

Diseño de datos

```
/** Fahr2Cel.c */

/** ñDiseo                                *
 * Representamos los grados con enteros *
 *                                       */
```

Signatura y declaración de propósito

```
/** Fahr2Cel.c */

/** ñDiseo                                  *
 * Representamos los grados con enteros  *
 ****************************************/

/** óDeclaracin                             *
 * Fahr2Cel(fahr: int) : void             *
 * Recibe la temperatura Fahrenheit y se *
 * retorna en grados Celsius              *
 *                                         *
```

Ejemplos

```
/** Fahr2Cel.c */

/** ñDiseo                                  *
 * Representamos los grados con enteros  *
 ****************************************/

 /** óDeclaracin                             *
 * Fahr2Cel(fahr: int) : void              *
 * Recibe la temperatura Fahrenheit y se  *
 * retorna en grados Celsius               *
 *                                         *
 * Ejemplos:                               *
 *                                         *
 * Entrada:         Salida:                *
 * 1                -17                    *
 * 20               -6                     *
 * 40                4                     *
 * 60                15                    *
 * 80                26                    *
 * 100               37                    *
 * 120               48                    *
 * 140               60                    *
 * 160               71                    *
 * 180               82                    *
 * 200               93                    *
 * 220               104                   *
 * 240               115                   *
 * 260               126                   *
 * 280               137                   *
 * 300               148                   *
 *                                         */
```

Definición del programa

```
/** Fahr2Cel.c */

#include<stdio.h>

/** Descripcion                          *
 * Representamos los grados con enteros  *
 ****************************************/

 /** Declaracion                          *
 * Fahr2Cel(fahr: int) : void            *
 * Recibe la temperatura Fahrenheit y se  *
 * retorna en grados Celsius             *
 *                                       *
 * Ejemplos:                             *
 *                                       *
 * Entrada:         Salida:              *
 * 1                -17                  *
 * 20               -6                   *
 * 40                4                   *
 * 60                15                  *
 * 80                26                  *
 * 100               37                  *
 * 120               48                  *
 * 140               60                  *
 * 160               71                  *
 * 180               82                  *
 * 200               93                  *
 * 220               104                 *
 * 240               115                 *
 * 260               126                 *
 * 280               137                 *
 * 300               148                 *
 ****************************************/

int fahr2Cel(int fahr)
{
        int celsius;
        celsius = 5 * (fahr-32) / 9;
        return celsius;
}

/** Declaracion                         *
 * main(void):int                       *
 * Inicia el programa y llamara a la    *
 * fahr2Cel una secuencia de veces      *
 * mostrando la tabla resultante        *
 ****************************************/

```

```
49  main()
50  {
51      int fahr, celsius;
52      int lower, upper, step;
53      lower = 0;      /* limite minimo de temperatura */
54      upper = 300;    /* limite ámximo */
55      step = 20;      /* ñtamao de rango */
56      fahr = lower;
57      while (fahr <= upper) {
58          celsius = fahr2Cel(fahr);
59          printf("%d\t%d\n", fahr, celsius);
60          fahr = fahr + step;
61      }
62  }
```

```
$ gcc -o fahr2cel fahr2cel.c
$ ./fahr2cel
0       -17
20      -6
40      4
60      15
80      26
100     37
120     48
140     60
160     71
180     82
200     93
220     104
240     115
260     126
280     137
300     148
$
```

2.2. Fibonacci

Problema Escribir un programa que calcule el n-nésimo término de fibonacci.

Diseño de datos

```
1  /** fibonacci.c */
2
3  #include <stdlib.h>
4  #include <stdio.h>
5
6  /** ñDiseo                                *
7   * La funcion sera recursiva y trabajara *
8   * con numeros enteros                    *
9   ****************************************/
```

Signatura y declaración de propósito

```
/** fibonacci.c */

#include <stdlib.h>
#include <stdio.h>

/** ñDiseo                                *
 * La funcion sera recursiva y trabajara *
 * con numeros enteros                   *
 ****************************************/

/** óDeclaracin                           *
 * fibonacci(k: int) : int               *
 * Recibe la posicion a calcular         *
 * retorna el numero de fibonacci        *
 *                                       *
```

Ejemplos

```
/** fibonacci.c */

#include <stdlib.h>
#include <stdio.h>

/** ñDiseo                                *
 * La funcion sera recursiva y trabajara *
 * con numeros enteros                   *
 ****************************************/

/** óDeclaracin                           *
 * fibonacci(k: int) : int               *
 * Recibe la posicion a calcular         *
 * retorna el numero de fibonacci        *
 *                                       *
 * Ejemplos:                              *
 *                                        *
 * Valor ingresado 8                      *
 * 0                                      *
 * 1                                          *
 * 1                                      *
 * 2                                      *
 * 3                                      *
 * 5                                      *
 * 8                                      *
 * 13                                     *
 *                                        */
```

Definición del programa

```c
/** fibonacci.c */

#include <stdlib.h>
#include <stdio.h>

/** ñDiseo                                  *
 * La funcion sera recursiva y trabajara *
 * con numeros enteros                    *
 ****************************************/

/** óDeclaracin                            *
 * fibonacci(k: int) : int                 *
 * Recibe la posicion a calcular           *
 * retorna el numero de fibonacci          *
 *                                         *
 * Ejemplos:                                *
 *                                          *
 * Valor ingresado 8                        *
 * 0                                        *
 * 1                                            *
 * 1                                        *
 * 2                                        *
 * 3                                        *
 * 5                                        *
 * 8                                        *
 * 13                                       *
 *                                          */

int fibonacci(int n) {

    printf(". ");
    if (n < 2)
        return n;
    else
        return fibonacci(n - 1) + fibonacci(n - 2);
                /* --- Llamado recursivo --- */
}

/** Declaracion                           *
 * main(void):int                         *
 * Inicia el programa y llamara a la     *
 * fibonacci una secuencia de veces      *
 * mostrando los valores resultantes     *
 ****************************************/
int main (){
    int k, fib=0;
    printf("\nfibonacci\n");
    printf("Ingrese numero:");
```

```
49      scanf("%d", &fib);
50      for(k=0; k < fib; k++) {
51          printf("k = %d : %d \n", k, fibonacci(k) );
52
53      }
54      return 0;
55  }
```

```
$gcc -o fibonacci fibonacci.c
$ ./fibonacci

fibonacci
Ingrese numero:8
. k = 0 : 0
. k = 1 : 1
. . . k = 2 : 1
. . . . . k = 3 : 2
. . . . . k = 4 : 3
. . . . . k = 5 : 5
. . . . . k = 6 : 8
. . . . . .
     . . k = 7 : 13
$
```

3. •

Referencias

1. Cátedra de Programación II: Ejercitación N° 2. Lic. Ciencias de la Computación. FCEyA. UNR (2016)
2. Kernighan, B., Ritchie, D.: The C Programming Language. Ansi C. Prince Hall Software Series, Second Edition (1988)