TP N°4: Macros y Funciones

La siguiente guía cubre los contenidos vistos en las clases teóricas:
6. Funciones y Metodologías de Diseño
7. Tareas del Preprocesador

A partir de esta guía algunos ejercicios estarán acompañados de un programa de testeo

Ejercicio 1

Decidir, en caso de ser posible, con qué valor termina cada una de las variables en los siguientes programas. En caso de haber errores de compilación, corregirlos.

a)

```
#define PI  3.1415

int
main(void)
{
      int a = 0;
      int b;

      b = PI + a++;

      return 0;
}
```

b)

```
#define PI 3.14

int
main(void)
{
      float a=0
      float b;

      b = a + PI++;

      return 0;
 }
```

c)

```
#define CUBO(x)  x * x * x

int cubo (int num);

int
main(void)
{
      int a=4, b, c;

      b = CUBO(a+1);
      c = cubo(a+1);

      return 0;
}

int
cubo ( int num )
{
      return num * num * num;
}
```

d)

```
#define CUBO(x)  ((x) * (x) * (x))

int cubo (int num);

int
main(void)
{
      int a=3, b, c, d, e, f;

      b = CUBO( ++a );
      c = CUBO( a++ );
      d = 3;
      e = cubo ( ++d );
      d = 3;
      f = cubo ( d++ );

      return 0;
}

int
cubo ( int num )
{
      return num * num * num;
}
```

e)

```
#define DIVISION(x, y, z)      {int i; \
                           for(z = 0, i = x-y; i >= 0; z++, i-= y); \
                               }
int
main( void )
{
      int a, b, c, m, n, p, x, y, z;

      a = 20; b=5; c=3;
      DIVISION(a, b, c);

      m=5; n=4; p=2;
      DIVISION(m++, n--, p);

      x=15; y=3; z=0;
      DIVISION(x, y, 0);

      return 0;
}
```

Ejercicio 2

Corregir, en caso de ser necesario, las macros del ejercicio anterior para que funcionen en forma correcta en todos los casos propuestos.

Ejercicio 3

Escribir, en no más de 5 líneas, un programa que defina y utilice una macro con un parámetro para calcular el volumen de una esfera. El programa deberá calcular el volumen para esferas de radios de 1 a 10 cm., e imprimir los resultados en forma tabulada.

Volumen de la Esfera = (4/3) * π * r^3

Ejercicio 4

Resolver el problema 4-14 del K&R. Utilizar la macro definida en un programa que realice intercambio de variables. (En no más de 5 líneas)

Exercise 4-14. Define a macro `swap(t,x,y)` that interchanges two arguments of type `t`. (Block structure will help.) □

Ejercicio 5

Escribir una macro MAXIMO2 que recibiendo tres parámetros, asigne al tercer parámetro el mayor de los dos primeros. Utilizarla en un programa para verificar su correcto funcionamiento. (En no más de 5 líneas)

Ejercicio 6

Escribir la macro MAXIMO3 para determinar el mayor de 3 valores, usando la macro definida en el ejercicio anterior. (En no más de 10 líneas). Verificar su funcionamiento invocándola desde un programa.

Ejercicio 7

Escribir la macro **isdigit** que dado un caracter indique si dicho caracter representa un dígito decimal o no.

Ejemplo de uso:

```
int letra;
letra = getchar();
if ( isdigit(letra) )
      printf("Es un dígito\n");
```

Ejercicio 8

Encontrar y corregir los errores en cada uno de los siguientes segmentos de programa.

a)

```
void
f1 (void)
{
      printf ("Dentro de la función f1\n");

      void
      f2(void)
      {
            printf ("Dentro de f2\n");
      }
}
```

b)

```
int
suma (int x, int y)
{
      int respuesta;
      respuesta = x + y;
}
```

c)

```
int
suma (int n);
{
      if (n<=0)
            return 0;
      else
            n + 1;
}
```

d)

```
void
f2( float a )
{
      float a;
```

```
      printf ("%f\n",a*a);
}
```

e)

```
void
f3 ( int letra )
{
      printf('Letra: ');
      putchar(letra);
      putchar("\n");
}
```

Ejercicio 9

La siguiente función recibe como parámetros de entrada las horas y los minutos que representan el momento de ingreso al trabajo de un empleado, y devuelve 1 (verdadero) si el empleado llegó a horario y 0 (falso) si llegó tarde. Las constantes ENT_HORA y ENT_MINUTOS están definidas fuera de la función e indican el horario esperado de entrada al trabajo. Testearla, y de no funcionar correctamente, corregirla.

```
int
llegaTemprano (const int hora, const int minutos)
{
      return (hora <= ENT_HORA && minutos <= ENT_MINUTOS);
}
```

Ejercicio 10

La siguiente función intenta ser una instrumentación del algoritmo de Euclides para calcular el MCD de dos números, pero no siempre da el resultado correcto. Indicar errores y corregirlos para que funcione correctamente.

```
int
dcm (int a, int b)
{
      int auxi;

      while (auxi>0)
      {
            a = b;
            b = auxi;
            auxi = a % b ;
      }

      return a;
}
```

Ejercicio 11

Escribir una función que reciba tres parámetros de entrada de tipo entero y que devuelva en su nombre el mayor de ellos. (En 2 líneas)

Ejercicio 12

Compilar y linkeditar cada par de códigos, indicando en cada caso si al ejecutarlos se obtiene el valor esperado y de no ser así explicar el motivo.

a)

```
#include <stdio.h>

int
main(void)
{
	int c = 2;

	c = neg(c);
	printf("%c\n",c);
	return 0;
}
```

```
int
neg(int n)
{
	return -n;
}
```

b)

```
#include <stdio.h>

int
main(void)
{
	double i = 5.42;

	i = neg(i);
	printf("%f\n",i);
	return 0;
}
```

```
double
neg(double n)
{
	return -n;
}
```

Ejercicio 13

Se desea crear un archivo ejecutable en base a dos archivos fuente: tp4_13a.c y tp4_13b.c. Corregir los errores en cada uno de los archivos fuente, indicando si son errores de preprocesamiento, compilación o linkedición.

Nota: no puede ser parte de la solución contar con un único archivo fuente.

Ejercicio 14

Idem ejercicio anterior pero los archivos se denominan tp4_14a.c y tp4_14b.c

Ejercicio 15

Idem ejercicio anterior pero los archivos se denominan tp4_15a.c, tp4_15a.h, tp4_15b.c, tp4_15b.h y tp4_15main.c

Ejercicio 16

Ejercicio de Parcial

Escribir la macro **DIVISOR** que reciba dos números enteros (no necesariamente positivos) y retorne 1 si el segundo es divisor del primero y cero si no es divisor. No usar funciones auxiliares.

Ejemplos de invocación:

```
int a, b = 10, c = 3;
a = DIVISOR(b,c); /* a = 0 */
a = DIVISOR(b,c-1);     /* a = 1 */
```

Ejercicio 17

Ejercicio de Parcial

Escribir la macro **ELAPSED** que al recibir dos medidas de tiempo en horas y minutos retorne la cantidad de minutos transcurridos. No usar funciones de la biblioteca estándar

Ejemplos de invocación:

```
int minutos, h1, h2, m1, m2;
h1 = 2; m1 = 10; h2 = 3; m2 = 15;
minutos = ELAPSED(h1, m1, h2, m2);        /* minutos = 65 */
minutos = ELAPSED(3, 15, 2, 10);          /* minutos = 65 */
minutos = ELAPSED(h1, m1, h1 + 1, m1);    /* minutos = 60 */
```