TP N°2: Expresiones en C - Entrada y Salida

Esta guía cubre los contenidos vistos en las clases teóricas:
3. Operadores y Expresiones
4. Entrada y salida de datos

A partir de esta guía se listan ejercicios de la bibliografía básica de la materia que corresponden a los contenidos vistos.
Se recomienda la lectura del libro **El lenguaje de programación C** Segunda Edición de Brian W. Kernighan y Dennis M. Ritchie y la resolución de todos los ejercicios del libro sugeridos en las guías.
Se encuentra disponible en Campus ITBA un documento con el listado de los capítulos seleccionados del libro y su correspondencia con las clases teóricas.

El código fuente de la biblioteca **getnum.c**, junto con su header, se encuentra disponible en Campus ITBA y en el Drive.
Para leer por entrada estándar valores de tipo entero usar la función ***getint***.
Para leer valores numéricos reales, usar la función ***getfloat*** o ***getdouble***.
Al invocar cualquiera de estas funciones, cuando el usuario ingrese un número seguido de caracteres extras, el número será aceptado y el resto del buffer será consumido.
Hay una variante de la biblioteca, el archivo **getnum2.c,** que contiene las mismas funciones que **getnum.c**, pero en caso de recibir caracteres extras después del número (excepto espacios) el mismo es rechazado como error.

Ejercicio 1

Decidir si en los siguientes pares de códigos cada una de las variables termina con el mismo valor numérico independientemente de la plataforma. Indicar en las evaluaciones el tipo de aritmética utilizada. **Justificar todas las respuestas.**

a)
```
int letra, flag;
flag = 0;
if ( letra == (int)'x')
        flag=1;
```

```
int letra, flag;
flag = 0;
if ( letra == 'x')
        flag=1;
```

b)
```
float x;
int y;
y = 5;
x = y / 3.0;
```

```
float x;
int y;
y = 5;
x = (float) (y / 3);
```

c)
```
float x;
int y;
y = 5;
x = y / 3.0;
```

```
int x;
int y;
y = 5;
x = (float) y / 3;
```

d)
```
int x;
float y;
y = 5.5;
x = y / 3;
```
```
int x;
float y;
y = 5.5;
x = (int) y / 3.0;
```

e)
```
double x;
int y;
x = 25;
y = x / 5;
```
```
double x;
float y;
x = 25.0 ;
y = x / 5;
```

f)
```
int x, y;
x = (char) 5;
y = x++;
```
```
int x, y;
x = 5.0;
y = x++;
```

g)
```
int i, j;
int k;
i = j = 20000;
k = i + j;
```
```
int i, j;
unsigned k;
i = j = 20000;
k = i + j;
```

h)
```
char a, b;
char c;
a = b = 60;
c = a + b;
```
```
char a, b;
unsigned char c;
a = b = 60;
c = a + b;
```

i)
```
char a, b;
char c;
a = b = 100;
c = a + b;
```
```
char a, b;
unsigned char c;
a = b = 100;
c = a + b;
```

j)
```
int x, y;
x = (char) 300;
y = x++;
```
```
int x, y;
x = 300.0;
y = x++;
```

k)
```
int x, y, z;
z = (x = 2) + (y = x)
```
```
int x, y, z;
x = y = 2;
z = x + y;
```

Revisá el código izquierdo del punto k). ¿Es posible saber el valor de z de antemano? ¿Con qué concepto de la teórica **3. Operadores y Expresiones** lo relacionás?

Ejercicio 2

a) Escribir un programa que imprima el mensaje "**Este es un programa en C**" en la salida estándar, colocando cada palabra en una línea. (Usar un único printf)

b) Redireccionar la salida del programa hacia el archivo **output.txt**. Una vez terminada la ejecución del programa, editar dicho archivo y verificar que contenga la salida correcta.

Ejercicio 3

¿Es válida la siguiente sentencia ?

```
printf("Hola que
            tal \n");
```

Ejercicio 4

Probar las siguientes instrucciones e investigar cuales son equivalentes entre sí. Sacar conclusiones.

```
printf("\a\n");
printf("%c\n", 7);
printf("%d\n", 7);
printf("%f\n", 7);
printf("%g\n", 7);
```

Ejercicio 5

Dada la siguiente declaración de variable:

char c = 't';

armar un programa que imprima en la salida estándar dicha variable, con los siguientes formatos: **%c, %d, %f.** Indicar qué valores imprime y de dónde obtiene dichos valores.

Ejercicio 6

Indicar qué salida se obtiene en cada ***printf*** del siguiente programa:

```c
#include <stdio.h>
#define palabra "ultima prueba"

int
main(void)
{
    int num1 = 53, num2 = 4;
    float num3 = 6.874;

    printf("num1= %10d\n", num1);
    printf("num2= %010d\n", num2);
    printf("num1= %-10d\n", num1);
    printf("num1+num2= %5i\n", num1 + num2);
    printf("num1+num2= %5f\n", num1 + num2);
    printf("num3= %-3.2f\n", num3);
    printf("num1= %-4d\nnum2= %-4d\nnum3= %3.1f\n", num1,num2, num3);
    printf("num3(como entero)= %d\n", num3);
    printf("num1 / num2 = %d\n", num1 / num2);
    printf("num2 / num1 = %d\n", num2 / num1);
    printf("esta es la %s\n", palabra);

    return 0;
}
```

Ejercicio 7

Dada la siguiente definición de variables

```
int edad = 25;
float longitud = 185.654;
char  letra = ´Z´;
```

imprimir en la salida estándar:

a) la variable ***edad***, encolumnada a izquierda, en un ancho de campo de 5 espacios.

b) la variable ***edad***, encolumnada a derecha, en un ancho de campo de 10 espacios.

c) la variable ***longitud***, encolumnada a la izquierda en un ancho de campo de 10 espacios, con 2 dígitos decimales.

d) la variable ***letra***, como número entero, en un ancho de campo de 8 espacios.

Ejercicio 8

Escribir un programa que lea una variable tipo ***float*** representando una velocidad en m/s, e imprima en la salida estándar el equivalente en km/h. **Utilizar una única variable.**

Para utilizar la función ***getfloat*** de la librería **getnum**, en el archivo fuente que estés utilizando para el ejercicio, necesitás incluir el archivo de encabezado correspondiente
#include "getnum.h"
Recordá que al ser una librería de inclusión (y no una perteneciente a la biblioteca estándar), el *include* se realiza con comillas (**"**lib.h**"**) y no con menor y mayor (<lib.h>)

Para obtener un ejecutable que utilice funciones de la librería **getnum** necesitás enviar **getnum.c** como un archivo fuente más a la invocación habitual de *gcc*. Por ejemplo:
$> gcc tp2ej8.c getnum.c -o tp2ej8 -Wall -pedantic -std=c99

Ejercicio 9

Leer un número entero positivo y escribir en la salida estándar el valor de dicho número dividido 2, usando decalaje.

Ejercicio 10

Escribir un programa que lea una variable entera que representa un intervalo de tiempo expresado en segundos. Realizar la conversión para imprimir en la salida estándar el equivalente en horas, minutos y segundos. **No utilizar variables auxiliares.**

Ejercicio 11

Leer dos caracteres desde la entrada estándar e imprimir en la salida estándar el de mayor valor ASCII, usando el operador condicional. Redireccionar la entrada de manera tal que ***getchar*** lea desde el archivo ***entrada.txt.*** Dicho archivo será escrito con un editor de texto sin formato y deberá tener el siguiente contenido:

a) **ABCD**

b) **A**
B
C
D

c) **A** **B** (usando tabulador entre A y B)

Indicar la salida generada en cada punto.

Ejercicio 12

Leer un caracter desde la entrada estándar e imprimir en la salida estándar si el mismo es una letra o no. Considerar que el conjunto de caracteres es el del lenguaje inglés, o sea ignorar la ñ y vocales acentuadas. Usar el operador condicional.

Ejercicio 13

Escribir un programa que lea dos enteros, e imprima si el segundo es múltiplo del primero. Usar el operador condicional.

Ejercicio 14

Escribir un programa que lea dos enteros y a continuación imprima el promedio, la suma, el menor y el mayor de ellos (o indicar que son iguales). Usar el operador condicional.

Ejemplo:

Si tuviésemos **int a, b;**
y se leyeran los valores **5** y **8**, se debería imprimir:

El promedio es 6.5
La suma es 13
El menor es 5
El mayor es 8
no son iguales

Ejercicio 15

Leer un carácter e imprimirlo en mayúscula. Considerar que el conjunto de caracteres es el del lenguaje inglés, o sea ignorar la ñ y vocales acentuadas. (Usar operador condicional).

Ejercicio 16

Leer dos caracteres seguidos de la entrada estándar e imprimir en la salida estándar si son iguales, si el primero es mayor o si el primero es menor, utilizando operadores condicionales:

a) desde teclado, hacia pantalla
b) redireccionando la entrada desde archivo, hacia pantalla
c) desde teclado, redireccionando la salida a un archivo
d) redireccionando la entrada y la salida con archivos

Ejemplo:

Si se ingresaran los caracteres **'a'** y **'g'**
se debería imprimir **El caracter 'a' es menor al caracter 'g'**

Ejercicio 17

Indicar, para aquellas expresiones que compilen correctamente, el valor que devuelve.

```
int
main(void)
{
      "3" + "4";
      '3' + '4';
      3 + 4;
      '3' + 4;
      3 + '4';
      "3" + '4';
      "3" + 4;
      return 0;
}
```

Ejercicio 18

Indicar si las siguientes afirmaciones son verdaderas o falsas, justificar

a) La función printf siempre empieza a imprimir al comienzo de una línea nueva.

b) El operador de módulo (%) puede ser usado sólo con operandos enteros

c) Las declaraciones de variable pueden aparecer en cualquier parte dentro de la función **main**.

d) Un programa C que imprime tres líneas de texto debe contener tres sentencias con **printf**

Ejercicio 19

Indicar cuál de los fragmentos de código cuenta con un mejor estilo de programación.

```
int c1 = getchar();
int c2 = getchar();
c1 > c2 ? printf("%c es mayor\n", c1) : printf("%c es mayor\n", c2);
```

```
int c1 = getchar();
int c2 = getchar();
printf("%c es mayor\n", c1 > c2 ? c1 : c2);
```

```
int c1, c2;
printf("%c es mayor", (c1 = getchar()) > (c2 = getchar()) ? c1 : c2);
```

Ejercicio 20

Realizar los siguientes ejercicios del libro de texto básico:
1.1, **1.2**, **1.6** y **1.7**

Exercise 1-1. Run the "`hello, world`" program on your system. Experiment with leaving out parts of the program, to see what error messages you get. □

Exercise 1-2. Experiment to find out what happens when `printf`'s argument string contains `\c`, where *c* is some character not listed above. □

Exercise 1-6. Verify that the expression `getchar() != EOF` is 0 or 1. □

Exercise 1-7. Write a program to print the value of `EOF`. □

Ejercicio 21

Analizar el programa tp2_21.c

a) Sin ejecutarlo decir qué va a imprimir.
b) Generar el ejecutable y verificar que se corresponda con el punto
c) Volver a generar el ejecutable pero agregando el flag
`-fsanitize=signed-integer-overflow`.
¿El resultado es el mismo que en el punto anterior? De no ser así, analizar por qué

Ejercicio 22

El siguiente programa funciona correctamente en algunas computadoras, pero no en otras. Indicar cuál puede ser el problema (si bien aún no se vió en detalle, con la instrucción while se puede ejecutar un ciclo mientras la condición sea verdadera, en este caso queremos leer todos los caracteres de la entrada estándar hasta que no haya más caracteres por leer)

```c
#include <stdio.h>

int
main(void)
{
     char c;
     while ((c = getchar()) != EOF)
          putchar(c);

    return 0;
}
```