

Ejercicio 1. Ejecute y compile el programa 1. ¿Cuál es el resultado?

Ni en el código ni en la practica se hace un intento por explicar la función “*getchar()*”, la cual está en el código y que aparentemente no funcionaba cuando lo probé.

Lo que hace este código es definir la variable “*x*” como un entero corto indefinido y “*manzanasT*” como un entero. Después de ello se define la variable “*x*” como “6” y define “*manzanasT*” como la operación “*5*x*”.

Después imprime en pantalla una cadena de caracteres (una oración). Primero se imprime un número entero, en este caso se imprime “*x*” como se especifica más adelante en el código, después se imprime “*pasteles requieren* ” seguido del valor (entero) de “*manzanasT*”.

Si ejecutáramos únicamente esta parte del código (suponiendo que después se ponga “*return 0*” y se cierren las llaves del main) nos imprimiría la frase “*6 pasteles requieren 30 manzanas*”.

Sin embargo hay dos funciones más en el código, el primero de ellos solo imprime “Introduce un carácter: ” y el segundo de ellos es “*getchar()*” y después de acababa el código sin explicación de lo que hace *getchar* ni de como usarlo.

De forma rápida, la función “*getchar*” sirve para guardar una letra en una variable. Sin embargo, tal y como esta definido en nuestro caso, no la guarda, debido a que necesitamos declarar una variable *char* y luego igualarla a la función *getchar* para que esta guarde dicha variable (letra). Por lo tanto, este primer código está mal.

```
// Programa para calcular manzanas para un pequeño fabricante de pasteles
/* Me di cuenta que utilizando dos lineas diagonales juntas puedo escribir
que se encuentren en una sola linea */

#include <stdio.h>

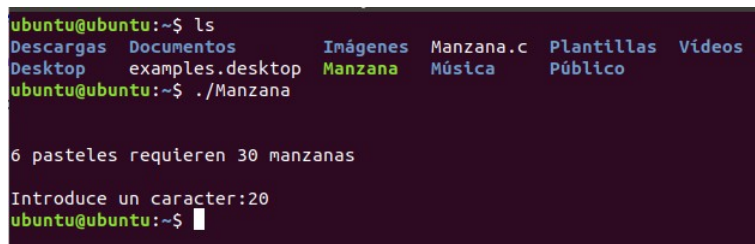
int main(void){

    unsigned short int x; //entero corto -0 a 255
    int manzanasT;
    x=6;
    manzanasT=5*x;

    printf("\n\n%d pasteles requieren %d manzanas\n\n", x, manzanasT);
    printf("Introduce un carácter:");

    getchar();
    return 0;
}
```

Lo que hace se intuye sencillamente: al crear el programa ejecutable y ejecutarlo lo que ocurrió fue que se imprimió en pantalla “6 pasteles requieren 30 manzanas”. Nota: Estoy dando por hecho que `\n` imprime un salto de linea. Y seguidamente se imprimió “Introduce un carácter: ”, sin embargo, al momento de introducir un carácter aleatorio con el teclado y darle “enter” terminó el programa y se cerró.



```
ubuntu@ubuntu:~$ ls
Descargas  Documentos  Imágenes  Manzana.c  Plantillas  Videos
Desktop    examples.desktop  Manzana  Música     Público

ubuntu@ubuntu:~$ ./Manzana

6 pasteles requieren 30 manzanas

Introduce un carácter:20
ubuntu@ubuntu:~$
```

Después de investigar un poco el funcionamiento de “*getchar()*” entendí que el ejemplo del profesor está mal, primero que todo porque debes declarar una variable de tipo “*char*”, en mi caso declaré la variable “*y*” y después en el cuerpo del código se iguala la variable de tipo *char* con la función “*getchar()*”, en mi código, por ejemplo, se ocupa así: “*y=getchar()*;”. Y también le agregué un “*printf*(“Has introducido el carácter: %c \n”, *y*);” para que me imprimiera el carácter que se introduzca. Por lo cual, mi código quedó de la siguiente manera:

```
// Programa para calcular manzanas para un pequeño fabricante de pasteles
/* Me di cuenta que utilizando dos líneas diagonales juntas puedo escribir
que se encuentren en una sola línea */

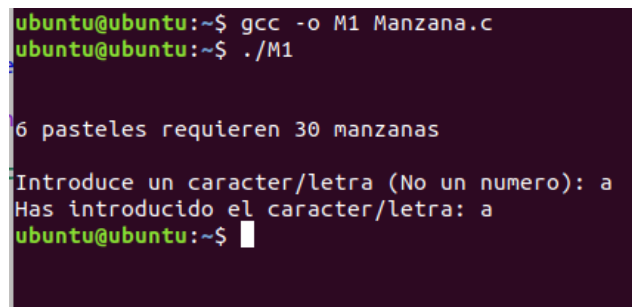
#include <stdio.h>

int main(void){

    unsigned short int x; //entero corto -0 a 255
    int manzanasT;
    char y;
    x=6;
    manzanasT=5*x;

    printf("\n\n%d pasteles requieren %d manzanas\n\n", x, manzanasT);
    printf("Introduce un caracter/letra (No un numero): ");
    y = getchar();
    printf("Has introducido el caracter: %c \n", y);
    return 0;
}
```

Para comprobar el código, solo se crea el ejecutable del código con el comando “*gcc*” y se ejecuta en la terminal. Lo que ocurrió es que se imprimió en pantalla dos saltos de línea seguido de “6 pasteles requieren 30 manzanas”, después otro salto de línea y “Introduce el carácter/letra (No un número): ” debido a que “*getchar()*” solo permite introducir un carácter alfabético. Y después se imprimió “Has introducido el carácter/letra: a”.



```
ubuntu@ubuntu:~$ gcc -o M1 Manzana.c
ubuntu@ubuntu:~$ ./M1

6 pasteles requieren 30 manzanas

Introduce un caracter/letra (No un numero): a
Has introducido el caracter/letra: a
ubuntu@ubuntu:~$
```

Ejercicio 2. Si ahora tenemos en cuenta que los pasteles también tienen sabor a manzana con canela por lo que cada pastel llevará 1.253 cc (centímetro cúbico) de esencia de canela.

¿Cómo modificará el programa anterior para obtener el total de esencia usada en dichos pasteles si el resultado final de la cantidad de esencia:

a) será expresada en litros?

Esto es sencillo, debido a que 1 cc es igual a 1 mililitro y esto es igual a 0,001 litros, solo debemos modificar el programa de tal manera que haga una conversión de unidades. Para hacer esto, primero declaramos una variable de tipo “*float*” que llamé “*esencia*” y después la definí como “1,253” que es el valor de la esencia de canela que utiliza un pastel en cc por el número de pasteles entre 1000. La primera parte calcula cuanta esencia de canela necesitan “*x*” pasteles, mientras que la división entre mil es lo que se debe hacer para convertir cc a litros.

Después, modificando un poco el “printf” para que también nos indique la cantidad en litros de esencia de canela utilizada en “x” manzanas, tenemos que el código quedó de la siguiente forma:

```
// Programa para calcular manzanas para un pequeño fabricante de pasteles
/* Me di cuenta que utilizando dos líneas diagonales juntas puedo escribir
que se encuentren en una sola línea */

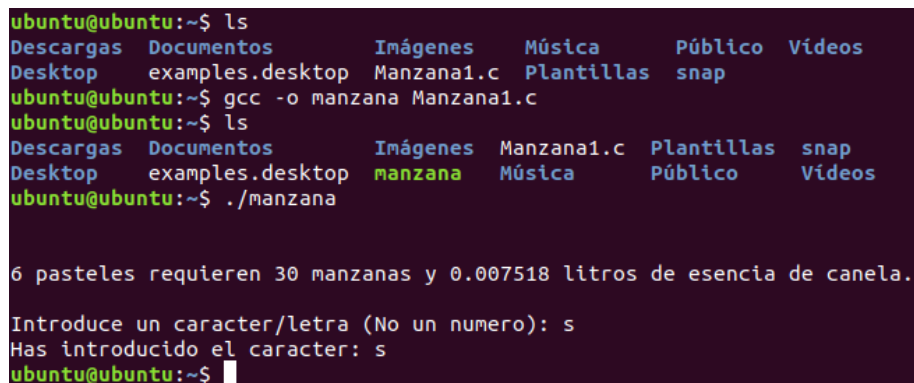
#include <stdio.h>

int main(void){

    unsigned short int x; //entero corto -0 a 255
    int manzanasT;
    char y;
    float esencia;
    x=6;
    manzanasT=5*x;
    esencia=(1.253*x)/1000;

    printf("\n\n%d pasteles requieren %d manzanas y %f litros de esencia de canela.\n\n", x,
manzanasT, esencia);
    printf("Introduce un caracter/letra (No un numero): ");
    y = getchar();
    printf("Has introducido el caracter: %c \n", y);
    return 0;
}
```

Creamos el programa ejecutable y lo ejecutamos para comprobar que, en definitiva, hicimos los cambios correctos. En este caso, la variable “x” esta definida como “x=6” por lo que se imprime que para 6 manzanas se necesitan 30 manzanas y 0,007518 litros de esencia de canela (7,518 ml).



```
ubuntu@ubuntu:~$ ls
Descargas  Documentos  Imágenes  Música  Público  Videos
Desktop    examples.desktop  Manzana1.c  Plantillas  snap
ubuntu@ubuntu:~$ gcc -o manzana Manzana1.c
ubuntu@ubuntu:~$ ls
Descargas  Documentos  Imágenes  Manzana1.c  Plantillas  snap
Desktop    examples.desktop  manzana  Música  Público  Videos
ubuntu@ubuntu:~$ ./manzana

6 pasteles requieren 30 manzanas y 0.007518 litros de esencia de canela.

Introduce un caracter/letra (No un numero): s
Has introducido el caracter: s
ubuntu@ubuntu:~$
```

También, indicar que no importa si cambiamos el valor de “x” pues esta no influye en el funcionamiento del programa (siempre y cuando x sea una variable de tipo entero corto), por eso si ocupamos “x=100” nos dice que para 100 pasteles se ocupan 500 manzanas y 0,1253 litros de esencia de canela (125,3 ml).

```
unsigned short int x; //entero corto -0 a 255
int manzanasT;
char y;
float esencia;
x=100;
manzanasT=5*x;
esencia=(1.253*x)/1000;
```

```

ubuntu@ubuntu:~$ gcc -o manzana Manzana1.c
ubuntu@ubuntu:~$ ./manzana

100 pasteles requieren 500 manzanas y 0.125300 litros de esencia de canela.

Introduce un caracter/letra (No un numero): a
Has introducido el caracter: a
ubuntu@ubuntu:~$ █

```

b) Como quedaría si el programa deba imprimir la cantidad de esencia total usada en cc hasta milésimos. Realice la modificación del programa anterior.

Utilizamos %9.3f para imprimir el valor del flotante asignado a la esencia de canela que imprima únicamente hasta los milésimos. Y también modificamos la definición de la esencia de canela para que ahora se de los valores en cc, por lo que la esencia queda expresada como “*esencia=(1.253*x)*”. Por lo que el código queda de la siguiente manera:

```

// Programa para calcular manzanas para un pequeño fabricante de pasteles
/* Me di cuenta que utilizando dos líneas diagonales juntas puedo escribir
que se encuentren en una sola línea */

#include <stdio.h>

int main(void){

    unsigned short int x; //entero corto -0 a 255
    int manzanasT;
    char y;
    float esencia;
    x=6;
    manzanasT=5*x;
    esencia=(1.253*x);

    printf("\n\n%d pasteles requieren %d manzanas y %9.3f cc de esencia de canela.\n\n", x,
manzanasT, esencia);
    printf("Introduce un caracter/letra (No un numero): ");
    y = getchar();
    printf("Has introducido el caracter: %c \n", y);
    return 0;
}

```

Aunque no entiendo por qué al ejecutar el programa se imprime un espaciado en medio del primer printf.

```

ubuntu@ubuntu:~$ gcc -o manzana Manzana2.c
ubuntu@ubuntu:~$ ./manzana

6 pasteles requieren 30 manzanas y      7.518 cc de esencia de canela.

Introduce un caracter/letra (No un numero): █

```

Editado 2 días después: Finalmente lo entendí, el primer valor indica cuantas cifras quieres mostrar antes del punto y el segundo cuantas cifras significativas quieres mostrar después del punto decimal, entonces como había puesto “9.3” esta mostrando el espacio de las otras 8 cifras (que valen 0) que están antes del 7, por eso esta ese espaciado entre “...manzanas y 7.518 cc...”.

El programa que corrige ese detalle es el siguiente:

```

// Programa para calcular manzanas para un pequeño fabricante de pasteles
/* Me di cuenta que utilizando dos líneas diagonales juntas puedo escribir
que se encuentren en una sola línea */

#include <stdio.h>

int main(void){

    unsigned short int x; //entero corto -0 a 255
    int manzanasT;
    char y;
    float esencia;
    x=6;
    manzanasT=5*x;
    esencia=(1.253*x);

    printf("\n\n%d pasteles requieren %d manzanas y %3.3f cc de esencia de canela.\n\n", x, manzanasT,
esencia);
    printf("Introduce un caracter/letra (No un numero): ");
    y = getchar();
    printf("Has introducido el caracter: %c \n", y);
    return 0;
}

```

Nótese que el único cambio fue colocar que el numero de cifras totales sea 3 igual que el numero de cifras decimales separadas por el punto (es decir, cambié %9,3f por %3,3f).

```

ubuntu@ubuntu:~$ gcc Manzana2.c
ubuntu@ubuntu:~$ ./a.out

6 pasteles requieren 30 manzanas y 7.518 cc de esencia de canela.

Introduce un caracter/letra (No un numero): 

```

Anexo aparte los tres programas que realice y utilice para estos ejercicios. Manzana.c es el código del programa del primer ejercicio, Manzana1.c es el código del programa del segundo ejercicio inciso a) y Manzana2.c es el código del programa del segundo ejercicio inciso b).