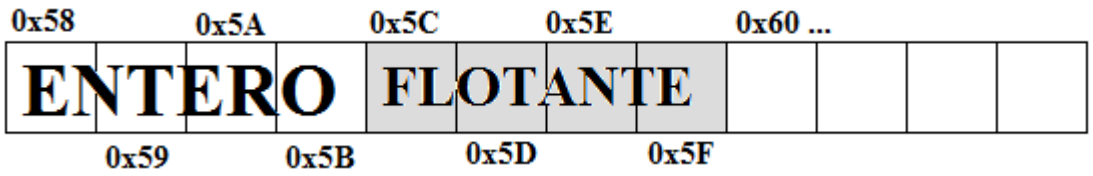
SISTEMAS OPERATIVOS Apuntadores y memoria RAM

1.- Declare en un programa las siguientes variables e imprima las direcciones de memoria en las cuales se encuentra almacenada cada una de ellas (para imprimir direcciones utilice el especificador de formato `%p`). Toda dirección se escribe siempre en hexadecimal y para indicarlo se antepone el símbolo `0x`.

```
char caracter = 0;
int entero = 0;
float flotante = 0;
long largo = 0;
double doble = 0;
```

A partir de la impresión y con ayuda de la función sizeof() dibuje en una hoja un mapa de memoria conteniendo las celdas, direcciones y variables almacenadas como muestra el siguiente ejemplo.



Utilice la función sizeof(véase con man) para verificar el tamaño de las variables.

2.- ¿Cuántos bits se almacenan en una dirección de memoria? ¿Cuál es el máximo valor hexadecimal, decimal y binario que puede almacenarse en una dirección de memoria? ¿Las variables se encuentran almacenadas en el orden en que fueron declaradas? ¿Las variables se almacenan en direcciones contiguas? ¿Un entero ocupa la misma cantidad de bytes en todas las computadoras?

3.- Observe de cuantos dígitos hexadecimales se componen sus direcciones de memoria al imprimirse. ¿Cuál sería la máxima dirección de memoria que se podría imprimir? Si cada dirección de memoria corresponde a un byte ¿Cuantos bytes sería posible almacenar en la RAM (escriba el prefijo k, M o T según corresponda)?

4.- ¿En su computadora cuántos bytes son necesarios para almacenar una dirección de memoria? Para verificarlo almacene una dirección de memoria en una de las cinco variables anteriores, e imprima su contenido para verificar que efectivamente se pudo almacenar.

5.- En realidad para almacenar direcciones de memoria se utiliza un conocidísimo tipo de variable denominada apuntador. Añada a continuación de las variables indicadas en el inciso uno, los siguientes apuntadores:

```
char *pcaracter;
int *pentero;
float *pflotante;
```

```
long *plargo;
double *pdoble;
```

Estas variables (tipo apuntador) también ocupan memoria. ¿Cuántos bytes ocupa cada apuntador? Complete el dibujo con el mapa de memoria que inició en la pregunta uno.

Como es de esperarse los valores contenidos en estas variables de tipo apuntador tienen al inicio basura, compruébelo al imprimir su contenido. Posteriormente inicialice el apuntador `pcaracter` con la dirección de la variable `caracter`, compruebe que lo ha hecho correctamente.

6.- Ahora declare el siguiente arreglo

```
char cadena[] = "ESCOM – IPN";
```

Mediante un ciclo `for`, imprima la dirección de cada uno de los caracteres que conforman dicha cadena. ¿Son direcciones continuas o discontinuas? ¿Que caracter contiene la dirección cadena+4? ¿Cómo demostraría en su programa que después del último caracter 'N' efectivamente se encuentra el caracter fin de cadena? (sugerencia, véase el manual del código ascii con man ascii).

7.- Inicialice una variable entera con el valor: 1234567890, posteriormente imprima el contenido de la variable en decimal y hexadecimal (utilice el especificador de formato `%x`). ¿Corresponde el valor decimal con su equivalente en hexadecimal? Ahora repita la operación pero inicialice la variable con el valor: -1234567890.

Para explicar porque no coincide en el caso de números negativos realice lo siguiente:

Pruebe a ver como se almacenan en RAM los siguientes valores: 0, 1, -1, 2, -2, 3, -3 …

Revise en Wikipedia el tema “Representación de números con signo“, y determine en su LAP como se almacenan los números con signo, haciendo conversiones a binario de los números hexadecimales (utilice la calculadora de UBUNTU en modo programador). Si ha entendido lo anterior responda ¿cuál es el máximo valor negativo y positivo que puede almacenar un entero en su LAP?, compruébelo asignando dichos valores a una variable.

8.- Imprima la dirección y el contenido de la memoria RAM en hexadecimal desde la dirección en la que inicia la declaración de variables y hasta la más alta posible byte por byte. Por ejemplo: `0xA34256-A9 0xA34257-F3 0xA34258-45 0xA34259-AD …`

¿Cuántos bytes le fue posible imprimir? ¿Qué sucedería si en lugar de imprimir en hexadecimal imprimiera con el especificador de formato %c? ¿Investigue por qué no es posible imprimir todo el contenido de la RAM instalada en su PC (desde 0x000…00000 hasta 0xFFFF…FFFF?