1. *Buscando señales en el espacio*

La estación *UPSETI* tiene la misión de investigar mensajes del espacio y requiere un programa que descifre un **mensaje** oculto que está llegando los días viernes en la madrugada. El mensaje es un conjunto de números de longitud variable. Esta longitud varía de **10 a 15.**

Por ejemplo el viernes 5 de junio llegó el siguiente mensaje de longitud 12

1	2	8	9	6	7	2	1	0	1	4	5
0	1	2	3	4	5	6	7	8	9	10	11

El viernes 19 de junio llegó el siguiente mensaje de longitud 15

9	8	5	2	1	0	3	2	1	4	7	8	9	0	4
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14

Conocedores de su habilidad para programar le solicitan construir un programa para descifrar los mensajes. Los expertos reportan que el mensaje tiene una secuencia de dígitos en forma de un **número capicúa de 4 cifras** en las últimas posiciones del mensaje. Con esta información el **programa debe controlar la cantidad de veces que encuentra este número capicúa**. Si el número capicúa aparece **5 veces debe imprimir "CONTACTO!**" y el programa debe terminar. Tener en cuenta que el programa funciona las 24 horas y los 365 días del año en búsqueda del mensaje oculto. *(ciclo repetitivo infinito)*

Por ejemplo este mensaje tiene el número capicúa en las últimas 4 posiciones del arreglo.

9	8	5	2	1	0	3	2	8	**8**	**7**	**8**
0	1	2	3	4	5	6	7	8	9	10	11

El programa deberá implementar, de forma obligatoria, las siguientes funciones:

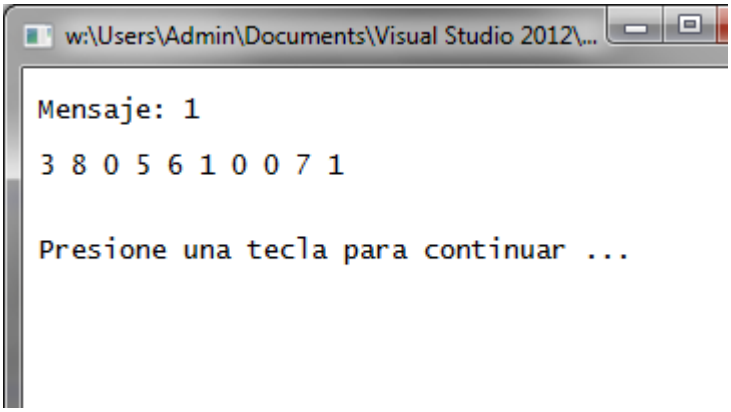
void Generar_Mensaje(int *Mensaje, int longitud_mensaje)
Esta función genera un arreglo de números enteros aleatorios en el rango de 0 a 9 y con una longitud que está señalada por el parámetro.

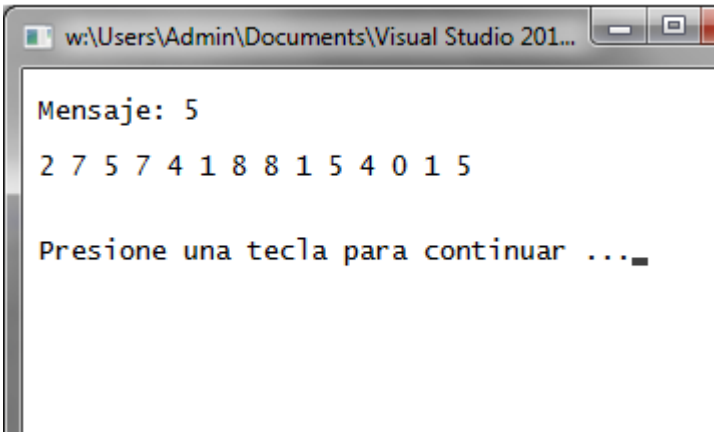
void Imprimir_Mensaje(int *Mensaje, int longitud_mensaje)
Esta función imprime el mensaje.

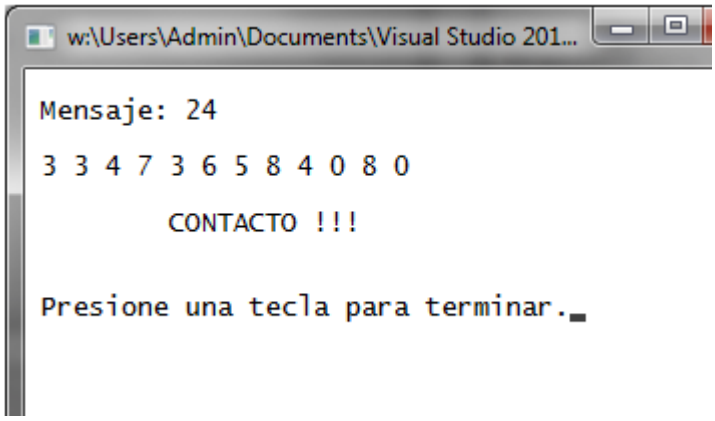
bool Existe_capicua(int *Mensaje, int longitud_mensaje)
Esta función recibe como parámetros un arreglo que representa el mensaje y la longitud del mismo. La función debe encontrar el número capicúa.

void main() que permite generar la cantidad de caracteres del mensaje (debe ser un valor entre **10 y 15**), generar el mensaje, imprimirla, mostrar la frecuencia de las vocales en la cadena y obtener y mostrar la longitud de la palabra de mayor tamaño contenida en la cadena. Para ello debe realizar el llamado correcto de las funciones generadas.

Ver un ejemplo en la siguiente página:


w:\Users\Admin\Documents\Visual Studio 2012\...
Mensaje: 1
3 8 0 5 6 1 0 0 7 1
Presione una tecla para continuar ...


w:\Users\Admin\Documents\Visual Studio 201...
Mensaje: 5
2 7 5 7 4 1 8 8 1 5 4 0 1 5
Presione una tecla para continuar ...


w:\Users\Admin\Documents\Visual Studio 201...
Mensaje: 24
3 3 4 7 3 6 5 8 4 0 8 0
CONTACTO !!!
Presione una tecla para terminar.

2. Un byte, que representa a un carácter, es una secuencia de 8 bits y un bit solo puede tomar el valor de 0 o 1. La información (conjunto de caracteres) confidencial que viaja por internet, suele ser protegida para evitar que cualquier persona pueda leerla. Una de las formas más seguras de proteger dicha información es encriptarla a un nivel de bits.

   Con este fin a continuación se describe un mecanismo para encriptar 1 byte:

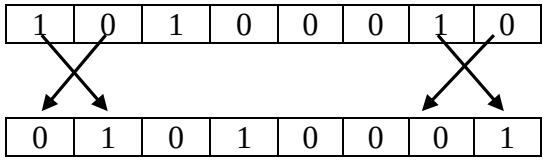
   - En primer lugar se debe tener como dato el byte a encriptar, representado como un arreglo de 8 bits

0	1	0	1	1	1	0	1

   - A continuación se prenden o apagan los bits, esto significa que si el bit es 1 se cambia por cero y viceversa.

1	0	1	0	0	0	1	0

   - Luego se intercambian los bits esto quiere decir el primero con el segundo, el tercero con el cuarto, el quinto con el sexto y el séptimo con el octavo.



   - Finalmente para visualizar el nuevo carácter encriptado, convertir la representación en bits a su equivalente en base decimal.

7	6	5	4	3	2	1	0
0	1	0	1	0	0	0	1

   Para convertir a la base decimal esta se calcula de la siguiente manera:

$$0*2^7+1*2^6+0*2^5+1*2^4+0*2^3+0*2^2+0*2^1+1*2^0 = 81$$

Conocido esto, se le solicita a usted que realice un programa en Lenguaje C++, en entorno consola y basada en funciones, que permita encriptar un byte, representado como una secuencia de 8 bits, según el mecanismo señalado.

Para realizar el programa deberá implementar, de forma obligatoria, las siguientes funciones:

void Generar_Vector(int Arreglo[])
Esta función genera un arreglo de 8 bits. Para generar el arreglo, recuerde que solo puede estar conformado por 0 o 1.

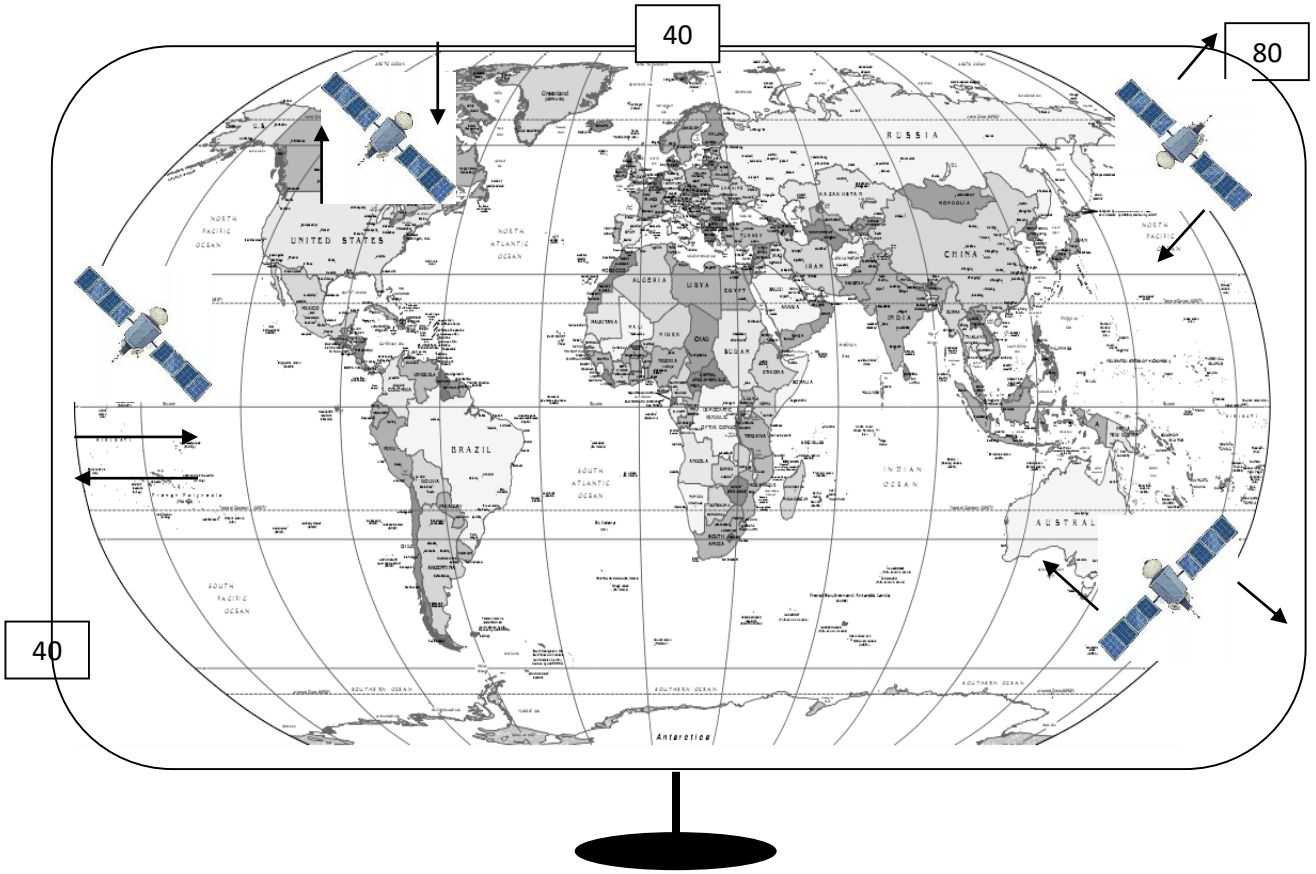
void prende_apaga_bits(int Arreglo[])
Esta función recibe como parámetro un arreglo de bits que representa un byte. Debe cambiar los bits del arreglo, es decir que si es uno lo cambia por cero o si es cero lo cambia por uno.

void intercambia_bits(int Arreglo[])
Esta función recibe como parámetro un arreglo de bits que representa un byte. Debe intercambiar los bits del arreglo, esto quiere decir el primero con el segundo, el tercero con el cuarto, el quinto con el sexto y el séptimo con el octavo.

int convertir_a_decimal (int Arreglo[])
Esta función entrega como resultado un entero que representa la conversión a base decimal del arreglo de bits.

void main() que permita generar el arreglo de bits y luego realice el llamado correcto de las funciones para generar e imprimir el nuevo carácter encriptado.

3. Se requiere simular el movimiento de 4 satélites sobre el globo terráqueo tal como se muestra en la figura:



Para ello, se debe dividir la pantalla por la mitad y en cada mitad deberán moverse dos satélites. Además **cada satélite será representado por un carácter y un color diferente**. El tamaño de la consola de ejecución representará las dimensiones del mapa esto significa que tiene 80 columnas por 40 filas.
Los satélites deben rebotar cada vez que lleguen a los bordes o a la mitad de la pantalla.

Para ello le solicita hacer un programa en Lenguaje C++, en entorno consola y basada en funciones, que realice la simulación del movimiento de los satélites.

Para realizar el programa deberá implementar, de forma obligatoria, las siguientes funciones:

void Ingresa_Posicion_Satelites(int PosicionXsatelite[], int PosicionYsatelite[], int dXsatelite[], int dYsatelite[])
Esta función ingresa la posiciones iniciales en X e Y de cada satélite así como sus desplazamientos tanto en X como Y. Debe validar que las posiciones iniciales deben encontrarse en el mapa. Además los desplazamientos solo pueden tomar el valor de -1, 0 o 1.

void Simulación(int PosicionXsatelite[], int PosicionYsatelite[], int dXsatelite[], int dYsatelite[])
Esta función realiza la simulación del movimiento de los satélites en cada una de las mitades. Este proceso finalizará cuando se presione una tecla.

void main() que realice el llamado correcto de las funciones anteriores.

Ejemplo:
Ingrese las Posiciones X de cada uno de los satélites: 1 15 70 65
Ingrese las Posiciones Y de cada uno de los satélites: 20 2 5 35
Ingrese el desplazamiento en X de los satélites: 1 0 -1 -1
Ingrese el desplazamiento en Y de los satélites: 0 1 1 -1

PosicionXsatelite	1	15	70	65
PosicionYsatelite	20	2	5	35
dXsatelite	1	0	-1	-1
dYsatelite	0	1	1	-1

4. Tomando como referencia el trabajo final del curso, se le solicita a usted que realice una aplicación, que desplace a un móvil sobre el mapa que se le proporciona a continuación (Figura 1).

   El jugador debe desplazar el móvil utilizando las teclas direccionales dejando un rastro por el camino que ha realizado. Debe tener en cuenta que el móvil no debe salir del tablero ni traspasar los muros ni su propio rastro.

   Por último, note que el jugador empieza su desplazamiento en el lado superior del tablero.

Para ello le solicita hacer un programa en Lenguaje C++, en entorno consola y basada en funciones, que realice el desplazamiento del móvil según las condiciones dadas. Para realizar el programa deberá implementar lo siguiente:

- Una función **genera_matriz**, que teniendo como parámetro una matriz de 40 x 40, la inicialice representando el tablero de la Figura 1.
- Una función **desplazar_movil**, que teniendo como parámetro una matriz de 40 x 40 y la ubicación inicial del móvil, permita desplazar el móvil utilizando las teclas direccionales dejando un rastro por el camino que ha realizado. Tome en cuenta que el móvil no debe salir del tablero ni traspasar los muros ni su propio rastro.
- La función **main** que solicite la ubicación inicial del móvil y luego realice el llamado correcto de las funciones anteriores para desplazar el móvil sobre el tablero dado. Recuerde validar la ubicación inicial del móvil.