**EXAMEN TEÓRICO 1ª EVALUACIÓN-PROGRAMACIÓN DAW**

1. Escribe una función en C++ que reciba como parámetro un array de 20 enteros y lo ordene por el algoritmo de la burbuja. **(1 punto)**

**void ordenBurbuja(int array[], int tam)**

**{**

**//Recorremos el array hasta el penúltimo elemento**

**for(int i=0; i < tam-1; i++)**

**{**

**//Se compara cada elemento con el siguiente**

**for(int j=0; j < tam-1; j++)**

**{**

**//Si vector[j]>vector[j+1] se intercambian**

**if(vector[j]>vector[j+1])**

**{**

**int aux=vector[j];**

**vector[j]=vector[j+1];**

**vector[j+1]=aux;**

**}**

**}**

**}**

**}**

1. Escribe una función que reciba como parámetros un array de enteros y un entero. Aplicando la búsqueda binaria, debe devolver la posición en la que el número se encuentra en el array o -1 si el número no se encuentra en el array. **(1 punto)**

**int busquedaBinaria(int vector[], int tam, int valor)**

**{**

**//Declaramos las variables que permiten hacer la busqueda binaria**

**int primera=0, ultima=tam-1, centro;**

**while( primera <= ultima)**

**{**

**//Obtenemos la posicion central**

**centro=(primera+ultima)/2;**

**if(valor==vector[centro])**

**{**

**//hemos encontrado el valor y devolvemos su posicion**

**return centro;**

**}**

**//Si el valor no esta en la posicion central seguimos buscando en la sublista izquierda o derecha**

**if(valor<vector[centro])**

**{**

**ultima=centro-1;**

**}**

**else**

**{**

**primera=centro+1;**

**}**

**}**

**//Si hemos llegado aqui es porque el valor no se encuentra en la lista**

**return -1;**

**}**

1. Escribe una función que reciba una matriz de enteros de NxMxP posiciones y un número entero y devuelva el número de veces que el número se encuentra en la matriz. **(1 punto)**

int repeticiones (int matriz[][][], int N, int M, int P, int valor)

{

int acum=0;

for(int i=0; i < N; i++)

{

for(int j=0; j < M; j++)

{

for(int k=0; k < P; k++)

{

if(matriz[i][j][k]==valor)

acum++;

}

}

}

return acum;

}

1. Algoritmo que lea por teclado una variable de tipo entero y utilizando un bucle WHILE lea por teclado tantos valores como indique la variable y muestre el valor mayor, el valor menor y la media de los valores leídos. La media será el retorno de la funcion y el número mayor y el menor se devolverán con parámetros de entrada/salida. **(2 puntos)**

float media(int \*mayor, int\* menor)

{

//Declramos la variable que lee el total de numeros a leer, la que va acumulando

// la suma y la que almacena cada uno de los numeros leidos

int total, num, acum=0;

//Inicializamos el valor del mayor y del menor

\*mayor=-9999;

\*menor=9999;

//Pedimos el total de numeros a lee

cout<<"Cuantos numeros se van a leer: ";

cin>>total;

//Vamos leyendo los numeros acumlando su suma para calcular la media

//Almacenando el menor y el mayor

int ind;

while(ind < total)

{

cout<<"Numero: ";

cin>>num;

//Sumamos el numero al acumulador

acum+=num;

//Si es menor que el valor almacenado en menor lo guardamos

if( num < \*menor)

{

\*menor=num;

}

//Si es mayor que el valor almacenado en mayor lo guardamos

if( num > \*mayor)

{

\*mayor=num;

}

//incrementamos el indice que controla e bucle

ind++;

}

return float(num)/total;

}

1. Función que reciba un número entero y devuelva el número multiplicado por 2 si es primo o devuelva el número multiplicado por 5 si no es primo. **(1 punto)**

boolean esPrimo(int numero)

{

//Declaramos una variable que toma el valor false si el numero no es primo

// true si lo es

boolean primo=true; //Suponemos que el numero es primo

//Analizamos si el número es primo. Un número es primo si solo es divisible

//entre él mismo y la unidad. Vamos obteniendo el cociente de dividir el

//numero entre todos los números menores a él. Si el resto de alguna de las

//divisiones es 0 el numero no es primo

for(int div=2; div < numero && primo; div++)

{

if( numero % div == 0)

{

primo=false;

}

}

//Si el numero es primo lo devolvemos multiplicado por 2

if( primo )

{

return numero\*2;

}

//Si el numero no es primo lo devolvemos multiplicado por 5

return numero\*5;

}

1. Función que reciba una cadena de caracteres como un array y elimine el primer carácter desplazando todos caracteres a la izquierda. **(1 punto)**

**//La cadena de caracteres se recibe como un array. Al ser una cadena de caracteres, la //ultima posicion del array es un carácter nulo por lo que no es necesario pasar el tamaño**

**// del array como parametro**

void eliminaPrimero(char cad[])

{

//Desplazamos las posiciones del array hacia la izquierda desde la segunda posición hasta

// la ultima

for(ind ind=0; cad[ind]; ind++)

{

cad[ind]=cad[ind+1];

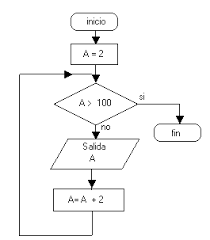
}

//Cuando se termina de ejecutar el bucle en las dos últimas posiciones del array hay un

//carácter nulo

}

1. Escribe el código correspondiente del siguiente diagrama de flujo utilizando: **(2 puntos)**
   1. Un bucle WHILE.
   2. Un bucle FOR



Solución con un bucle while

int a=2;

while( a <= 100)

{

cout<<a;

a\*=2;

}

Solución con un bucle FOR

for( int a=2; a <= 100; a\*=2)

{

cout<<a;

}

1. Dado el siguiente segmento de código: **(1 punto)**

char cadena[20], car;

int ind=0;

do{

cin>>car;

if ( car != ‘\n’)

{

cadena[ind]=car;

ind++;

}

}while(car != ‘\n’);

cout<<cadena;

Indica el proceso que ejecuta el código y cuál sería su salida

El algoritmo pretende leer por teclado los caracteres a almacenar en una cadena de caracteres de 20 posiciones hasta que se introduzca un salto de linea. Se observan los siguientes errores:

* Solo se realiza una lectura antes del bucle, por lo que solo se almacenará el primer carácter leído y el bucle será infinito. Es un error de ejecución.
* No se ha cerrado la cadena de caracteres con un carácter nulo por lo que la salida de cout es indeterminada. Es un error de ejecución
* El bucle while no controla que los caracteres leídos sean menos que 20, por lo que se puede desbordar el array.