

IMPLEMENTACIÓN DEL EXÁMEN EXPLORATORIO

1.- ¿Cual es la diferencia entre las siguientes variables?

int *a,b; 'a' guarda la direccion en memoria de la variable tipo entero a la que vaya a apuntar.

'b' es una variable tipo entero.

int **c; 'c' es un apuntador que podrá apuntar a otro apuntador, el cual, a su vez, podrá apuntar a una variable entera.

float *d; 'd' es una variable tipo apuntador que podrá ubicar la posición en memoria de un flotante.

2.- Dado un apuntador **int *a;** y un entero **#define N 56**

- Asignar memoria para obtener un vector de N elementos
- Llenar todo el vector con enteros POSITIVOS aleatorios
- Escribir el código que encuentra el número más grande sin importar la longitud del vector
- Escribir el código para encontrar el segundo número más grande sin importar la longitud del vector (nota: es posible cambiar el contenido del vector si es necesario)
- escribir en pantalla los 2 números encontrados
- liberar la memoria

```
/* Guillermo AG: Programa para el inciso dos del examen exploratorio.  
Enero, 2011. Buscador de los dos numeros mas grandes de un grupo */
```

```
#include <stdio.h>
```

```
#include <stdlib.h>
```

```
#include <time.h>
```

```
#define N 56
```

```

int main()
{
    int *a,i,primero,segundo;

    a=malloc(N*sizeof(int)); // Asignación de memoria

    if (a==NULL) {system("cls"); printf("\n\nNo hay memoria asignada a 'a'");
        getch(); exit(0);} // Muestra si es que no se asignó memoria

    system("cls");printf("\n\nNumeros aleatorios\n\n");
    primero=0;segundo=0;
    for(i=0;i<N;i++){a[i]=rand();
        if(primero<=a[i]){segundo=primero;primero=a[i];}
        else {if(segundo<a[i])segundo=a[i];}
        printf("[%i] = %i \t",i+1,a[i]); // Imprime cada numero asignado
    } /* Asigna eneteros positivos aleatorios y va obteniendo los valores
        mayores en cada iteración */

    printf("\n\nSe ha creado un vector de %i elementos; de entre ellos, el "
        "mayor es %i y el segundo mayor es %i\n\n",N,primero,segundo);
    // Impresion de resultados

    free(a); // Liberación de memoria
    getch();
    return(0); // Fin exitoso
}

```

3.- Dado float **a; y el entero int M=14;

- Darle memoria dinámica para contener una matriz de MxM
- Llenar toda la matriz de ceros
- Dar el código para llenar la tridiagonal principal con números aleatorios entre 10 y 20.
- Decir el numero de entradas que contiene un numero impar
- liberar toda la memoria de la matriz

/* Guillermo AG: Programa para el inciso tres del examen exploratorio.
 Enero, 2011. Tridiagonal de una matriz cuadrada con numeros aleatorios
 entre 10 y 20.*/*

```

#include <stdio.h>
#include <stdlib.h>
#define M 14

int main()
{
    float **a;
    int i,j,k,contador=0;

    a=malloc(M*sizeof(float)); // Asignación de memoria al vector de columnas
    if (a==NULL) {system("cls"); printf("\n\nNo hay memoria asignada a 'a'");
        getch(); exit(0);} // Muestra si es que no se asignó memoria

    for(i=0;i<M;i++){a[i]=malloc(M*sizeof(float));
        if (a[i]==NULL){for (j=0;j<i;j++){free(a[j]);}free(a);
            printf("\n\nError");getch();exit(0);}
    }
    // Hasta aquí, se ha creado la matriz de M x M.

    for(i=0;i<M;i++){for(j=0;j<M;j++){a[i][j]=0;}}
    // Inicializacion a ceros de la matriz

    for(i=0;i<M;i++){j=rand()%11+10;k=j%2;a[i][i]=j;if(k==1)contador++;
        if(i!=0){j=rand()%11+10;k=j%2;a[i][i-1]=j;if(k==1)contador++;}
        if(i<M-1){j=rand()%11+10;k=j%2;a[i][i+1]=j;if(k==1)contador++;}
    } // Asigna los aleatorios (10-20) a la tridiagonal

    system("cls");
    printf("\n\nNumeros aleatorios entre el 10 y el 20 en la "
        "tridiagonal de una matriz de %i x %i con ceros en "
        "los demas elementos; de entre ellos,"
        " hay %i impares.\n\nLa matriz ha sido:\n\n",M,M,contador);

    for(i=0;i<M;i++){for(j=0;j<M;j++){if(a[i][j]==0)printf(" ");
        printf(" %g ",a[i][j]);}printf("\n");} // Imprime la matriz usada

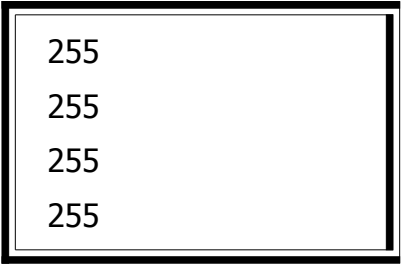
    for (i=0;i<M;i++){free(a[i]);}free(a); // Liberación de memoria
    getch(); return(0); // Fin exitoso
}

```

4.- ¿Qué sale a pantalla?

```
unsigned int *b;  
char c[4];  
b = (unsigned int*) c;  
*b = 4294967295; //2^32 - 1;  
  
printf("%d\n",c[0]); printf("%d\n",c[1]); printf("%d\n",c[2]); printf("%d\n",c[3]);
```

Salida a pantalla:



```
255  
255  
255  
255
```

El 255 es el máximo entero sin signo que se puede almacenar en un espacio de 1 byte (8 bits, $2^8 - 1$). Y en 4 bytes se ha almacenado el máximo entero sin signo que posible: 4294967295 ($=2^{32} - 1$), llenando con 1 cada bit o con F cada cuatro bits. Después, se toman los espacios reservados en memoria de 1 byte (tipo char) y nos queda el byte lleno de 1's (o FF en hexadecimal).

IMPLEMENTACIÓN DEL ALGORITMO DE EUCLIDES PARA ENCONTRAR EL MCD

// GAG enero 2011: Algoritmo de Euclides para encontrar el MCD

```
#include <stdio.h>
#include <stdlib.h>
#include <math.h>

int residuo(int x, int y);

int main()
{ int a,b,c,d,e,f=1;

  system("cls");
  printf("\n\n\tPrograma que calcula el Maximo Comun Divisor"
        " de dos numeros enteros\n\n"
        "Ingresa los dos enteros con un 'enter':\n\n");scanf("%i%i",&b,&c);

  b=abs(b);c=abs(c);      // Toma solo valores no negativos
  a=(b+c+abs(c-b))/2; d=a;    // Seleccion del mayor
  b=(b+c-abs(c-b))/2; e=b;    // Seleccion del menor

  // Calculo con el operador % modulo
  c=a%b;
  do{ if (c==0) {printf("\n\nEl MCD es %i utilizando el operador modulo.",b);
    getch(); f=0;}
    else { a=b; b=c; c=a%b;}
  }while (f==1); // Ciclo sin fin

  // Calculo sin el operador % modulo
  f=1; c=residuo(d,e);
  do{ if (c==0) {printf("\n\nEl MCD es %i haciendo restas.",e); getch(); f=0;}
    else { d=e; e=c; c=residuo(d,e);}
  }while (f==1);

  return(0);
}

int residuo(int x, int y)
{ while(x-y>=0) x-=y;
  return(x);
}
```