Nombre:	
C.I.:	

TECNÓLOGO EN INFORMÁTICA

PRINCIPIOS DE PROGRAMACIÓN

Solución **EXAMEN 22/02/10**

Ejercicio 1 (10 puntos)

Llene los espacios en blanco:

1) Cada programa C inicia su ejecución en la función _____main_____.
2) Con una _____{_____ se inicia el cuerpo de cada función y con una _____}_____ se termina.
3) Cada instrucción termina con _____Punto y coma_____.
4) La instrucción _____if_____ sirve para tomar decisiones.
5) La función _____printf_____ se utiliza para imprimir en pantalla.

Ejercicio 2 (15 puntos)

Que imprimen los siguientes programas:

a)
```
int main()
{
        int count=1;
        while (count <= 10)
                printf("%d\n", count++);
}
```

Sol:

1
2
3
4
5
6
7
8
9
10

Nota: Luego del while la variable count queda con valor 11.

b) int main()

```
{
        int count=0;
        while (count <= 5)
                imprimo(++count);
}

void imprimo(int c)
{
        if  ((c%2) == 0)   printf("%d\n", c);
        else printf("%d\t",c);
}
```

Sol:

La instrucción while se ejecuta 6 veces, imprime:

Salida:

1 2
2 4
5 6(enter al final)

Ejercicio 3 (20 puntos)

Escriba una función que lea el tamaño de los lados de un cuadrado e imprima con asteriscos y espacios en blanco un cuadrado vacio de ese tamaño como en el ejemplo:

```
****   (ej.: cuadrado de lado 4)
*¨   *
*    *
****
```

```
int  dibujaCuadrado(){
   int lado=0;
   printf("Ingrese el lado del cuadrado a dibujar: ");
   scanf("%d",&lado);
   int ch,cv;
  ch=cv=0;//variables contador horizontal y vertical, para contar los /
/caracteres
  for(ch=0;ch<lado;ch++)printf("*");//lado de arriba
  printf("\n");
   ch=0;
   for(cv=1;cv<lado-1;cv++){
         //voy dibujando por fila
        printf("*");//lado izquierdo
        for(ch=1;ch<lado-1;ch++)
          printf(" ");//espacio
        printf("*\n");//lado derecho
  }
  for(ch=0;ch<lado;ch++)printf("*");//lado de abajo


}
```

Ejercicio 4 (15 puntos)

a) Escriba una función que lea un entero mayor o igual a cero (debe verificarse esta condición, en caso de no cumplirse debe retornar cero la función) y retorne su factorial.

```
int main(){
     int n;
     printf("Ingrese un entero mayor o igual a cero");
     scanf("%d",&n);
     if(n<0) return 0;
     else{
         return factorial(n);
     }

}

int factorial(int j){
    if(j<2) return 1;
    else return j*factorial(j-1);
}
```

b) Escriba un programa que utilizando dicha función aproxime el valor de la constante **e** empleando la formula:

e= 1 + 1/1! + 1/2! + 1/3! + ……+1/10!

```
int e10(){
    int temp=0;
    for(int i=1;i<11;i++)
      temp=temp+1/factorial(i);

}
```

Ejercicio 5 (40 puntos)

Se desea construir un pequeño programa para la administración de un club deportivo. La cantidad de socios que podrá soportar el sistema estará acotada por la constante **MAX_SOCIOS**. En una primera etapa, y en lo concerniente a esté ejercicio, la aplicación a construir debe desplegar un menú principal con las siguientes opciones.

1-Ingresar Socio
2-Borrar Socio
3-Listar Socio
4-Salir

Los datos que interesan del socio son el nombre y la cédula.

Se pide

a) Escriba en C/C++, la declaración de la estructura para representar a los socios en el sistema. El nombre con el que se identificara a la estructura debe ser Socio.

(Asuma que el largo de la cédula es siempre de **MAX_CAR_CI** caracteres, y que el nombre no puede contener más de **MAX_NOMBRE** caracteres)

```
struct Socio{
        char ci[MAX_CAR_CI+1]; // se agrega un lugar para el character nulo
        char NOM[MAX_NOMBRE+1]; // se agrega un lugar para el character nulo


};
```

b) Implemente la función
void ingresarSocio(struct Socio socios[],int indice)

Que pide al usuario mediante un mensaje en pantalla el nombre y la cédula, y lee del teclado estos valores, agregando el nuevo socio en el arreglo **socios**, en la posición dada en **índice**.

```
void ingresarSocio(struct Socio socios[],int indice){
        printf("Por favor ingrese su nombre");
        scanf("%s",socios[indice].nom);//automáticamente se agrega el carácter /0 al
//final
        printf("Por favor ingrese su cédula");
        scanf("%s",socios[indice].ci);
}
```

c) Implemente la función
void borrarSocio(struct Socio socios[],char cedula[])

Que elimina del arreglo **socios** la posición donde coincida la cedula del socio con la **cedula** pasada cómo parámetro. En caso de que la **cédula** pasada como parámetro no se encuentre en el arreglo **socios**, este permanecerá inalterado. Para indicar que un elemento en el arreglo **socios** es vació, el campo cedula debe contener la cadena 'Vacio'.

```
void borrarSocio(struct Socio socios[],char cedula[]){
  bool encontre=false;
  int cont=0;
  while(¡encontre && cont< MAX_SOCIOS){
        if(strcmp(cedula,socios[cont].ci)==0){
            encontre=true;
             strcpy(socios[cont].ci,"VACIO");
         }else{
            cont++;
         }
   }
}
```

d) Implemente la función
void imprimirSocios(struct Socio socios[])

Debe imprimir en pantalla todos los socios (nombre y cédula) del arreglo socios.

Tener en cuenta que en el arreglo pude haber posiciones donde no haya socios válidos. Un Socio en el arreglo es vacío cuando en el campo cédula de la estructura se encuentra el valor “Vacío”.

```
void imprimirSocios(struct Socio socios[]){
 for(int i==0;i<MAX_SOCIOS;i++){
       if(strcmp(“VACIO”,socios[i].ci)!=0){
           printf(“Nombre : %s\n”, socios[i].nom);
           printf(“CI : %s\n\n”, socios[i].ci);

       }
   }

}
```

e) Implemente el programa principal, donde el sistema le del teclado las opciones 1, 2, 3 y 4 y realiza las funciones pertinentes. En caso de que la opción seleccionada sea salir, el sistema desplegará el mensaje “Fin Programa.!!!Que la pase Bien!!!” en la pantalla. Puede asumir que las entradas ingresadas al sistema por el teclado son válidas. Utilice las funciones implementadas en los pasos anteriores.

```
#define MAX_SOCIOS   …
#define MAX_CAR_CI   …
#define MAX_NOMBRE   …

int main(){
    struct Socio socios[MAX_SOCIOS];
    //inicializo el arreglo
    for(int i=0;i<MAX_SOCIOS;i++){
        strcpy(socios[i].ci,”VACIO”);
     }

  int tecla=5;
  int poslibre=-1;//utilizada para saber las posiciones libres del arreglo socio
  scanf(“%d”,&tecla);
  do{
  switch(tecla){
          case 1:  poslibre=hayLugar(socios);
                   if(poslibre!=-1)
                        ingresarSocio(socios,poslibre);
                   else
                        printf(“No pueden ingresar más socios”);
                   break;
           case 2: borrarSocio(socios); break;

           case 3: listarSocios(socios);break;

           case 4: break;//tecla vale 4
          default: tecla=5;


  }
```

```
}while(tecla!=4)


}

//función auxiliar hay lugar

int hayLugar(struct Socio socios[]){
   int lugar=-1;
  bool encontre=false;
  int cont=0;
  while(¡encontre && cont< MAX_SOCIOS){
       if(strcmp(“”,socios[cont].ci)==0){
           encontre=true;
           lugar=cont;
        }else{
           cont++;
        }
   }

  return lugar;
}
```