PROGRAMACIÓN I

TRABAJO PRÁCTICO N° 3

Funciones, variables simples y estructuras

DEPARTAMENTO DE ELECTRICIDAD, ELECTRÓNICA Y COMPUTACIÓN

AÑO 2018

Contenido

Problemas Resueltos

Funciones con variables simples, que utilizan parámetros por valor

Resuelto con una función que usa parámetros por valor y que retorna valor

Ingresar un número real *base* y un número entero positivo *exponente* presentar por pantalla el valor de *base* elevado al valor de *exponente* . Utilizar una función para realizar el cálculo de *base* elevado a *exponente*. Utilizando variables simples.

```
#include <stdio.h>
//Declaración de funciones
float potencia(float x, int y); //x e y son los parámetros por valor

int main(int argc, char *argv[]){
	float base,p;
	int exponente;
	printf("Ingresar la base (nro. real): ");
	scanf("%f",&base);
	do{
		printf("Ingresar el exponente (nro. entero positivo): ");
		scanf("%d",&exponente);
	}while(exponente<=0);
	p=potencia(base,exponente); //Argumentos base y exponente se corresponden con los parámetros y p copia el valor retornado por la función
	printf("El valor de %.2f elevado a %d es = %.2f\n\n",base,exponente,p);
 return 0;
}
//Definición de funciones
float potencia(float x, int y){
	float p=1;
	int c=1;
	while(c<=y){
		p=p*x;
		c++;
	}
return p;
}
```

Resueltos adicionales, con una función que No usa parámetros y otra función que Si usa parámetro por valor

Ingresar un número entero positivo, luego presentar por pantalla un mensaje que indique si el número ingresado es o no es primo.

```
#include <stdio.h>

//DECLARACION DE FUNCIONES
int entero_positivo();
int primo(int x);
```

```
int main(int argc, char *argv[]){
	int x,b;
	x=entero_positivo();
	b=primo(x);
	if(b==2)
		printf("El numero %d SI es primo\n\n",x);
	else
		printf("El numero %d NO es primo\n\n",x);
return 0;
}

//DEFINICION DE FUNCIONES
/*
	Recibe: nada
	Retorna: un entero positivo ingresado por el usuario
 */
int entero_positivo() {
	int x;
	do{
		printf("Ingresar número entero positivo: ");
		scanf("%d",&x);
	}while(x<=0);
return x;
}

/*
	Recibe: un valor entero x
	Retorna: retorna un entero b, si b=2 el valor x es primo
 */
int primo(int x){
	int d=1,b=0;
	while(d<=x){
		if(x%d==0){
			b++;
		}
		d++;
	}
return b;
}
```

Ingresar un número entero *num*. Utilizar una función que indique si es positivo (´P´), negativo (‘N’) o cero (‘Z’), devolviendo el carácter correspondiente.

```
#include <stdio.h>
//DECLARACIÓN DE FUNCIONES
char evaluar(int x);

int main(int argc, char *argv[]){
	int num;
	printf("Ingresar un número entero: ");
```

```
	scanf("%d",&num);

	if(evaluar(num)=='P')
		printf("El numero %d es positivo\n\n",num);
	else
		if(evaluar(num)=='Z')
			printf("El numero %d es cero\n\n",num);
		else
			printf("El numero %d es negativo\n\n",num);
return 0;
}

//DEFINICION DE FUNCIONES
char evaluar(int x) {
	char e='Z';
	if(x>0)
		e='P';
	if(x<0)
		e='N';
return e;
}
```

Ingresar dos números A y B, enteros positivos en formato octal (controlar mediante una función que cada número posea formato octal y que sea positivo), luego realizar la suma directa de ambos números (mediante otra función). Presentar los números ingresados y el resultado.

```
#include <stdio.h>

int octal_positivo();
int suma_octal (int a, int b);

int main(int argc, char *argv[]){
	int a,b,r;
	a=octal_positivo();
	b=octal_positivo();
	r=suma_octal(a,b);
	printf("La suma en octal de %d + %d es = %d",a,b,r);
return 0;
}

int octal_positivo(){
	int x,aux,d,b=0;
	do{
		printf("Ingresar un número positivo en formato octal:");
		scanf("%d",&x);
		aux=x;
		if(x<=0)
			b=1;
```

```
                while(x>0){
                        d=x%10;
                        x=x/10;
                        if(d>7){
                                b=1;
                                x=0;
                        }
                }
        }while(b==1);
return aux;
}

int suma_octal(int a, int b){
        int s,d1,d2,ac=0,r=0,p=1;
        while(a>0||b>0||ac>0){
                d1=a%10;
                d2=b%10;
                a=a/10;
                b=b/10;
                s=d1+d2+ac;
                if(s>7){
                        s=s-8;
                        ac=1;
                }
                else
                        ac=0;
                r=r+s*p;
                p=p*10;
        }
return r;
}
```

Ingresar un número entero positivo (realizar el control que sea positivo con una función), luego presentar por pantalla un mensaje que indique si dicho número es un número triangular (realizar la verificación mediante una función).

```
#include<stdio.h>
int positivo();
int triangular(int y);
int main(int argc, char *argv[]){
        int x,b;
        x=positivo();
        b=triangular(x);
        if(b==0)
                printf("El número %d SI es triangular",x);
        else
                printf("El número %d NO es triangular",x);
```

```
return 0;
}

int positivo(){
	int x;
	do{
		printf("Ingresar un número entero positivo:");
		scanf("%d", &x);
	}while(x<=0);
return x;
}

int triangular(int y){
	int i=1,b=0,t;
	do{
		t=i*(i+1)/2;
		i++;
	}while(t<y);
	if(t!=y)
		b=1;
return b;
}
```

Funciones que utilizan variables simples, estructuras y parámetros por valor

Resuelto con una función que usa parámetros por valor y que retorna valor utilizando una estructura

Ingresar dos números enteros, luego presentar por pantalla el resultado de la diferencia entre el primer número ingresado y el segundo entero ingresado, además presentar en pantalla una letra en mayúscula que indique si el resultado de la diferencia es positivo 'P', negativo 'N' o cero 'Z'. Utilizar una función que retorne el resultado de la diferencia con la letra correspondiente empleando una variable tipo estructura.

```
#include <stdio.h>

//Definición de la estructura
typedef struct{
	int resta;
	char letra;
	} salida;

//Declaración de funciones
salida diferencia(int x, int y); //Retorna una estructura de tipo salida

int main(int argc, char *argv[]){
	salida difletra; //La variable difletra es de tipo salida
	int a,b;
	printf("Ingresar el 1er nro.: ");
	scanf("%d",&a);
	printf("Ingresar el 2do nro.: ");
	scanf("%d",&b);
	difletra=diferencia(a,b);//a y b se corresponden con x e y respectivamente
```

```
	printf("\nEl valor de %d menos %d es = %d\n\n",a,b,difletra.resta);//Se accede al valor de la diferencia escribiendo difletra.resta
	printf("La letra del signo de la diferencia es %c\n\n",difletra.letra);//Se accede al contenido de letra escribiendo difletra.letra
  return 0;
}

//Definición de funciones
salida diferencia(int x, int y){
	salida res;//res es una estructura del tipo salida
	res.resta=x-y;
	if (x-y>0){
			res.letra='P';
	}
	else{
			if(x-y<0){
				res.letra='N';
			}
			else res.letra='Z';
	}

  return res;//La variable res contiene la diferencia y la letra
}
```

Resuelto con una función que usa parámetro por valor tipo estructura y que retorna valor utilizando una variable simple tipo caracter

Ingresar un número entero y un número real. Luego presentar por pantalla una letra en mayúscula que indique si el resultado del producto del número entero por el real es positivo 'P', negativo 'N' o cero 'Z'. Utilizar una función que utilice parámetro por valor tipo estructura y que retorne un caracter.

```
#include <stdio.h>
//Definición de la estructura
typedef struct{
		int entero;
		float real;
	} entrada;

//Declaración de funciones
char producto(entrada entero_real); //Retorna un caracter

int main(int argc, char *argv[]){
	entrada entreal;
	char letra;
	printf("Ingresar un nro. entero: ");
	scanf("%d",&entreal.entero);
	printf("Ingresar un nro. real: ");
	scanf("%f",&entreal.real);
	letra=producto(entreal);
	    printf("La letra correspondiente al signo del producto entre %d y %.2f es %c\n\n",entreal.entero,entreal.real,letra);
```

```
	return 0;
}

//Definición de funciones
char producto(entrada entero_real){
	char letra;
	if (entero_real.entero*entero_real.real>0){
		letra='P';
	}
	else{
		if(entero_real.entero*entero_real.real<0){
			letra='N';
		}
		else letra='Z';
	}

	return letra;
}
```

Resuelto con una función que usa parámetros por valor tipo estructura y que retorna valor utilizando una estructura

Ingresar un número entero y un número real. Luego presentar por pantalla el producto de los números ingresados y una letra en mayúscula que indique si el resultado del producto es positivo 'P', negativo 'N' o cero 'Z'. Utilizar una función que utilice parámetro por valor tipo estructura y que retorne una estructura que contenga el producto y la letra correspondiente.

```
#include <stdio.h>

//Definición de las estructuras
typedef struct{
	int entero;
	float real;
	} entrada;

typedef struct{
	float producto;
	char letra;
	} salida;


//Declaración de funciones
salida producto(entrada entero_real); //Utiliza un parámetro de tipo entrada y retorna una estructura de tipo salida

int main(int argc, char *argv[]){
	entrada x;
	salida y;
```

```
        //x e y son estructuras de tipo entrada y salida respectivamente
        printf("Ingresar un nro. entero: ");
        scanf("%d",&x.entero);
        printf("Ingresar un nro. real: ");
        scanf("%f",&x.real);
        y=producto(x);
        printf("\nEl producto de %d por %.2f es = %.2f\n",x.entero,x.real,y.producto);
       printf("La letra correspondiente al producto es %c\n\n",y.letra);

        return 0;
}

//Definición de funciones
salida producto(entrada entero_real){// la estructura x se corresponde con entero_real
        salida s; //s es una estructura de tipo salida
        s.producto=entero_real.entero*entero_real.real;
        if (s.producto>0){
                    s.letra='P';
        }
        else{
                    if(s.producto<0){
                                s.letra='N';
                    }
                    else s.letra='Z';
        }

return s;
}
```

Resuelto con una función que usa parámetros por valor tipo estructura y que retorna valor utilizando una estructura

Realizar un programa que permita ingresar N números (realizar el control de que N sea positivo), por cada número ingresado se debe tener en cuenta además el orden en que fue ingresado, se se debe utilizar una estructura para contener el número ingresado y el orden. Luego utilizar una función para encontrar el menor valor de los números ingresados, indicando el orden en que ingresó. Presentar por pantalla los resultados.

```
#include <stdio.h>

//Definición de la estructura
typedef struct{
        int numero;
        int orden;
        } estructura;

//Declaración de funciones
estructura mayor(estructura ent, estructura aux, int c); //Retorna una estructura de tipo
```

```
estructura

int main(int argc, char *argv[]){
	estructura may, entrada;//se utilizan dos variables tipo estructura
	int n,c;
	c=1;
	printf("Ingresar cantidad de números: ");
	scanf("%d",&n);
	while (c<=n){
		printf("Ingresar un nro. entero: ");
		scanf("%d",&entrada.numero);//se registra el valor ingresado en la estructura
		entrada.orden=c;//se registra el orden de ingreso en la misma estructura
		may=mayor(entrada,may,c);//Se utiliza la misma variable may para conservar los
cambios
		c++;
	}
	printf("\nEl mayor número ingresado es %d y el orden en que ingreso es
%d\n",may.numero,may.orden);
return 0;
}

//Definición de funciones

estructura mayor(estructura ent, estructura aux, int c){
	if (ent.orden==1)
		aux=ent;//se inicializa como mayor al primer valor ingresado con su
correspondiente orden
	else{
		if(ent.numero>aux.numero){
			aux=ent;
		}
	}
return aux;
}
```

Funciones con variables simples que usan parámetros por referencia

Resuelto con una función que usa parámetros por valor, parámetro por referencia y que retorna valor

Ingresar un número entero y un número real, luego presentar por pantalla la suma y el producto de dichos números.

```
#include <stdio.h>
//Declaración de funciones
float suma_producto(int x, float y, float *sum); //sum es un parámetro por referencia

int main(int argc, char *argv[]){
```

```
	int entero;
	float real,suma,producto;
	printf("Ingresar un nro. entero: ");
	scanf("%d",&entero);
	printf("Ingresar un nro. real: ");
	scanf("%f",&real);

	producto=suma_producto(entero,real,&suma); //Notar el agregado de & en la variable suma
	printf("\nEl valor de %d más %.2f es = %.2f\n\n",entero,real,suma); //El valor de suma retorna modificado por ser parámetro por referencia
	printf("El valor de %d por %.2f es = %.2f\n\n",entero,real,producto);

 return 0;
}

//Definición de funciones
float suma_producto(int x, float y, float *sum){ //sum apunta a suma
	float p;
	p=x*y;
	*sum=x+y; //sum en un puntero o apuntador a la variable suma
	//todos los cambios realizados utilizando *sum se efectúan directamente en suma
	return p;
}
```

Resuelto con una función que usa parámetros por valor, parámetro por referencia y que No retorna valor

Ingresar un número entero y un número real, luego presentar por pantalla la suma y el producto de dichos números.

```
#include <stdio.h>
//Declaración de funciones
void suma_producto(int x, float y, float *sum, float *pro); //sum y pro son parámetros por referencia

int main(int argc, char *argv[]){
	int entero;
	float real,suma,producto;
	printf("Ingresar un nro. entero: ");
	scanf("%d",&entero);
	printf("Ingresar un nro. real: ");
	scanf("%f",&real);

	suma_producto(entero,real,&suma,&producto); //Notar el agregado de & en la variable suma y en producto
	printf("\nEl valor de %d más %.2f es = %.2f\n\n",entero,real,suma); //El valor de suma retorna modificado
	printf("El valor de %d por %.2f es = %.2f\n\n",entero,real,producto); //El valor de producto retorna modificado
```

```
 return 0;
}

//Definición de funciones
void suma_producto(int x, float y, float *sum, float *pro){ //sum apunta a suma y pro a producto
	*sum=x+y;//los cambios realizados utilizando *sum se efectúan directamente en suma
	*pro=x*y;//los cambios realizados utilizando *pro se efectúan directamente en producto

	return;//No retorna valor
}
```

Ingresar dos caracteres (dos letras por ejemplo), luego presentar por pantalla dichos caracteres, pero ordenados en forma ascendente, o un mensaje en caso de que sean iguales.

```
#include <stdio.h>
#include <stdlib.h> //para usar system()

int ordenar (char *a, char *b);

int main()
{
  char x,y;//x e y contendrán los caracteres infgresados
  int b;
  system("clear");//en Linux (cls en Windows)
  printf("Ingresar un caracter:");
  scanf("%c",&x);//scanf asigna el caracter ingresado a la variable x
   fgetc(stdin); //elimina el enter (que también es un caracter) que se ingresó después de la letra, de lo contrario y tomará dicho enter
  printf("Ingresar otro caracter:");
  scanf("%c",&y);

  b=ordenar(&x,&y);

  if(b==0)
  {
      printf("\n\nLos caracteres ordenados son: %c, %c\n\n",x,y);
  }
  else
  {
      printf("\n\nLos caracteres son iguales...\n\n");
  }
  return 0;
}
```

```
int ordenar (char *a, char *b)//a y b apuntan a x e y respectivamente
{
  char c;
  int d=0;
  if(*a>*b)//al utilizar *a y *b se accede indirectamente  a x e y respectivamente
  {
      c=*a;//utilizando *a la variable c toma el valor de x
      *a=*b;//el valor de x copia el valor de y
      *b=c;//la variable y copia el valor de c
  }
  else
  if(*a==*b)//se compara el valor de las variables x e y
      d=1;
  return d;
}
```

Ingresar N números enteros positivos y formar un nuevo número colocándolos en forma consecutiva. Por ejemplo: N = 5 y se ingresan los siguientes valores 12 , 0 , 97, 0 , 501. La salida será: 120970501

```
#include <stdio.h>

void numComp ( int x, int *nroComp);

int main(int argc, char *argv[]){
       int x, i, n, dig;
       int nroComp; nroComp = 0;

       do {
              printf("\n Ingresar la cantidad de valores N : ");
              scanf("%d", &n);
       } while ( n <= 0 );

       for ( i = 1; i<= n; i++){
              printf(" Ingrese un valor x: ");
              scanf("%d", &x);
              numComp(x, &nroComp);
       }

       printf("\n%d", nroComp);
       printf("\n\n");
return 0;
}
void numComp( int x, int *nroComp){
       int aux;
       int pot;
       aux = x;
       if (aux != 0){
```

```
                pot = 1;
                while (aux >0){
                        pot = pot * 10;
                        aux = aux / 10;
                }
        }
        else
                pot = 10;
        *nroComp = *nroComp * pot + x;

return;
}
```

Funciones que utilizan estructuras y usan parámetros por referencia

Problema resuelto con una función que usa parámetro por referencia tipo estructura y que retorna valor

Ingresar dos caracteres (dos letras por ejemplo), luego presentar por pantalla dichos caracteres, pero ordenados en forma ascendente, o un mensaje en caso de que sean iguales.

```
#include <stdio.h>
//Definición de las estructuras
typedef struct{
        char primera;
        char segunda;
        } referencia;

//Declaración de funciones
int ordenar(referencia *e);//la función utiliza un parámetro tipo estructura llamado e por
referencia y retorna una variable simple

int main(int argc, char *argv[]){
        referencia est;
        int iguales;
        printf("Ingresar un caracter: ");
        scanf("%c",&est.primera);//se ingresa el primer caracter en la estructura
        fgetc(stdin);//Limpia el buffer del teclado
        printf("Ingresar otro caracter: ");
        scanf("%c",&est.segunda);//se ingresa el segundo caracter
        printf("\nLos caracteres ingresados son: %c, %c\n",est.primera,est.segunda);
        iguales=ordenar(&est);//el parámetro e que es un puntero a estructura, apunta a la
estructura est
        if (iguales==0){
                printf("\nLos caracteres son iguales\n");
          }
          else{
                printf("\nLos caracteres ordenados son: %c, %c\n",est.primera,est.segunda);
          }
```

```
	return 0;
}

//Definición de funciones
int ordenar(referencia *e){//e es un puntero a la estructura est
	char aux;
	int i=1;
	if(e->primera == e->segunda){//Para acceder a un campo en una estructura por referencia, se utiliza el operador flecha -> en lugar del operador punto
//en este caso se compara el valor del primer campo de est con el segundo campo de est
		i=0;
	}
	else{
		if(e->primera > e->segunda){ //se compara el primer caracter contenido en la estructura est con el segundo caracter contenido en la misma estructura
		  aux = e->primera;//aux copia el primer caracter de est
		  e->primera = e->segunda;//el primer campo de est copia el valor del segundo campo de est
		  e->segunda = aux;//el segundo campo de est copia el valor de aux
		}
	}
	return i;
}
```

Problema anterior resuelto con acceso a la estructura empleando puntero y utilizando * y . en lugar de ->

```
#include <stdio.h>
//Definición de las estructuras
typedef struct{
	char primera;
	char segunda;
	} referencia;

//Declaración de funciones
int ordenar(referencia *e);//la función utiliza un parámetro tipo estructura llamado e por referencia y retorna una variable simple

int main(int argc, char *argv[]){
	referencia est;
	int iguales;
	printf("Ingresar un caracter: ");
	scanf("%c",&est.primera);//se ingresa el primer caracter en la estructura
	fgetc(stdin);//Limpia el buffer del teclado
	printf("Ingresar otro caracter: ");
```

```
        scanf("%c",&est.segunda);//se ingresa el segundo caracter
        printf("\nLos caracteres ingresados son: %c, %c\n",est.primera,est.segunda);
        iguales=ordenar(&est);//el parámetro e que es un puntero a estructura, apunta a la
estructura est
        if (iguales==0){
              printf("\nLos caracteres son iguales\n");
          }
          else{
              printf("\nLos caracteres ordenados son: %c, %c\n",est.primera,est.segunda);
          }

        return 0;
}

//Definición de funciones
int ordenar(referencia *e){//e es un puntero a la estructura est
        char aux;
        int i=1;
        if((*e).primera == (*e).segunda){//Para acceder a un campo en una estructura por
referencia, también se puede utilizar el operador * y el punto
//en este caso se compara el valor del primer campo de est con el segundo campo de est
                i=0;
        }
        else{
                if((*e).primera > (*e).segunda){ //se compara el primer caracter contenido
en la estructura est con el segundo caracter contenido en la misma estructura
                  aux = (*e).primera;//aux copia el primer caracter de est
                  (*e).primera = (*e).segunda;//el primer campo de est copia el valor del
segundo campo de est
                  (*e).segunda = aux;//el segundo campo de est copia el valor de aux
                }
        }
        return i;
}
```

Problema resuelto con función que utiliza parámetro por referencia tipo estructura

De un grupo de N personas, se registra peso, altura, fecha de nacimiento y la inicial del nombre de cada una. Implementar un programa que muestre los datos de la persona que tenga el menor peso y determinar también el promedio de las alturas ingresadas. Utilizar las funciones y estructuras que sean necesarias.

```c
#include <stdio.h>
typedef struct{
	int dia;
	int mes;
	int anio;
	}fecha;

typedef struct{
	float peso;
	float altura;
	fecha nacimiento;
	char inicial;
} persona;

//Declaración de funciones
persona ingresar();//función que permite ingresar los datos en una estructura y luego retornar dicha estructura
void buscarpeso(persona per, int c, float *menor_peso, persona *p);//función que recibe datos de una persona en per y asigna en la estructura p los datos de la persona con menor peso
float alturas(persona per, float suma_alturas);//función que recibe datos de una persona en per y acumula los datos de alturas
void mostrar(persona p);//función que recibe datos de una persona en p y muestra su contenido

int main(int argc, char *argv[]){
	persona per, per_menor_peso;
	int N,c;
	float suma_alturas, prom_alturas, menor_peso;
	suma_alturas=0;
	c=0;
	printf("Ingresar la cantidad de personas: ");
	scanf("%d",&N);

	while(c<N){
		per=ingresar();
		buscarpeso(per, c, &menor_peso, &per_menor_peso);
		suma_alturas=alturas(per,suma_alturas);
		c=c+1;
	}

	prom_alturas = suma_alturas/N;

	printf("\n\nLos datos de la persona con el menor peso son los siguientes:\n");
	mostrar (per_menor_peso);
```

```
	printf("\n\nEl promedio de las alturas es: %.2f\n", prom_alturas);
return 0;
}
//Definición de funciones
persona ingresar(){ //Función que permite ingresar datos de personas
	persona p;
	printf("Ingresar el peso: ");
	scanf("%f",&p.peso);
	printf("Ingresar la altura: ");
	scanf("%f",&p.altura);
	printf("Ingresar dia de nacimiento: ");
	scanf("%d",&p.nacimiento.dia);
	printf("Ingresar mes de nacimiento: ");
	scanf("%d",&p.nacimiento.mes);
	printf("Ingresar anio de nacimiento: ");
	scanf("%d",&p.nacimiento.anio);
	printf("Ingresar la inicial del nombre: ");
	fgetc(stdin);
	scanf("%c",&p.inicial);
return p;
}

void buscarpeso(persona per, int c, float *menor_peso, persona *p){
	if (c==0){
		*menor_peso=per.peso;
		(*p)=per;
	}
	else{
		if(per.peso<*menor_peso){
			*menor_peso=per.peso;
			(*p)=per;
		}
	}
return;
}


float alturas(persona per, float suma_alturas){
	suma_alturas=suma_alturas+per.altura;
return suma_alturas;
}

void mostrar(persona p){//Función que muestra el contenido de la estructura
	printf("\nPeso: %.2f",p.peso);
	printf("\nAltura: %.2f",p.altura);
	printf("\nDia de nacimiento: %d",p.nacimiento.dia);
	printf("\nMes de nacimiento: %d",p.nacimiento.mes);
```

```
      printf("\nAnio de nacimiento: %d",p.nacimiento.anio);
      printf("\nInicial del nombre: %c",p.inicial);
return;
}
```

Ámbito de las variables y clases de almacenamiento

Problema Resuelto con variable static

Ingresar N números enteros positivos y formar un nuevo número colocándolos en forma consecutiva. Por ejemplo: N = 5 y se ingresan los siguientes valores 12 , 0 , 97, 0 , 501. La salida será: 120970501

```
#include <stdio.h>
long int nroComp(int);

int main(int argc, char* argv[]){
long int nro;
int x, n, c = 1;
do{
printf("\n Indique cuantos numeros va a ingresar: ");
scanf("%d", &n);
} while (n<=0);

for( ; c<=n ; c++){//el valor de c está inicializado en 1 en la declaración
do{
printf("\n Ingresar un numero : ");
scanf("%d", &x);
} while (x<0);
nro = nroComp(x);
}
printf("\n %ld", nro);
printf("\n\n");
return 0;
}

long int nroComp(int x){
static int nro;// la variable entera nro es estática, se inicializa por defecto en cero
```

```
sólamente la primera vez que se invoca a la función. Luego conserva su valor cada vez que
finaliza la función nroComp
int pot10 = 1, aux;
aux = x;
if( x == 0 )
pot10 = 10;
while (aux >0){
aux = aux / 10;
pot10 = pot10 * 10;
}
nro = nro * pot10 + x;

return nro;//en la siguiente llamada a la función nroComp, la variable nro contendrá este
último valor (ya no comenzará desde cero)
}
```

Ingresar N números enteros, determinar el mayor número ingresado y su posición. Utilizando estructura static

```
#include <stdio.h>
//Definición de las estructuras
typedef struct{
        int x;
        int orden;
        } estructura;

 //Declaración de funciones
estructura mayor(estructura ent); //Retorna una estructura

int main(int argc, char *argv[]){
        estructura m, e;
        int n,c;
        c=1;
        printf("Ingresar cantidad de números: ");
        scanf("%d",&n);
        while (c<=n){
                printf("Ingresar un nro. entero: ");
                scanf("%d",&e.x);
                e.orden=c;
                m=mayor(e);
                c++;
        }
        printf("\nEl mayor numero ingresado es %d y el orden en que ingreso es
%d\n",m.x,m.orden);
return 0;
```

```
}

//Definición de funciones

estructura mayor(estructura ent){
	static estructura aux;//la variable tipo estructura aux es estática, por lo tanto conserva los valores que tienen sus campos, cada vez que finaliza la función mayor
	if (ent.orden==1)
		aux=ent;
	else{
			if(ent.x>aux.x){
				aux=ent;
			}
	}
return aux;//en la siguiente llamada a la función, la variable aux comenzará con los valores que posea al momento de retornar
}
```

Funciones recursivas

Problemas resueltos

Escribir un programa que permita ingresar un número entero no negativo y que muestre el valor del factorial de dicho número utilizando el método recursivo.

```
#include<stdio.h>

int factorial(int x);//Declaración o prototipo de la función recursiva

int main(){
int x,y;
printf("Ingresar un número entero no negativo:");
scanf("%d",&x);
y=factorial(x);
printf("El factorial de %d es %d",x,y);
return 0;
}


int factorial(int x){//Definición de la función recursiva
if(x>1)
return(factorial(x-1)*x);//si el valor de x es > 1, la función se invoca a si misma enviando como argumento al valor de (x-1)
else return 1; //sólamente cuando x sea igual a 1 retornará el valor 1
} //la función factorial finalmente retornará el valor de 1*... *(x-3)*(x-2)*(x-1)*x
```

Problemas Recomendados (Obligatorio)

Con funciones que utilizan variables simples y/o estructuras y parámetros por valor.

1. Implementar un programa que permita ingresar un número entero no negativo en formato binario (realizar el control que sea no negativo y que tenga formato binario con una función), presentar por pantalla el número ingresado pero expresado en decimal (realizar la conversión con otra función).

2. Codificar un programa en el que se solicita el ingreso de un caracter, luego presentar por pantalla un mensaje que indique si dicho caracter es un dígito, una letra del alfabeto en mayúscula, una letra del alfabeto en minúscula o de otro tipo de caracter. Se debe utilizar funciones.

3. Implementar un programa que permita calcular la diferencia de volumen entre dos cilindros circulares utilizando funciones. Los datos disponibles de cada cilindro son: radio de la base y la altura, utilizar una función para calcular el volumen de cada cilindro y que use variables tipo estructura para contener los datos de entrada. Presentar por pantalla el resultado.

4. Codificar un programa que permita el ingreso de un número entero positivo *numerador* y otro número entero positivo *denominador* (utilizando una función para controlar que sea positivo cada uno de los números), utilizar una estructura que contenga dichos números y luego mediante otra función que recibe la estructura ingresada, calcular el cociente y el resto de la división entera utilizando el método de restas sucesivas. La función debe retornar una estructura que contenga el resultado del cociente y del resto. Presentar por pantalla los resultados.

5. Implementar un programa que permita determinar si un medicamento se encuentra o no vencido, conociendo la fecha de vencimiento y la fecha actual. Utilizar una función que realice el cálculo y use variables tipo estructura que puedan contener los datos de año, mes y día para cada fecha. Presentar por pantalla el mensaje correspondiente. Utilizar parámetros por valor tipo estructura.

6. Codificar un programa en el que se solicita el ingreso de dos números complejos (ingresando parte real y parte imaginaria de la forma binómica en una variable tipo estructura para cada número complejo), luego utilizar una función para calcular el cociente de los mismos, además debe retornar el resultado utilizando una variable tipo estructura. Presentar por pantalla los números ingresados y el resultado.

7. Codificar un programa en el que se solicita el ingreso del peso y la talla de N personas, utilizando una estructura para contener peso y talla, luego determinar, utilizando una función que use parámetros por valor tipo estructura y que retorne una estructura, los datos de la persona que tenga la mayor talla. Presentar por pantalla los resultados.

8. Realizar un programa que permita ingresar N números (realizar el control mediante una función que N sea positivo), por cada número ingresado se debe tener en cuenta además el orden en que

fue ingresado, entonces por cada número que se ingresa se debe guardar el mismo con su orden en una estructura. Luego utilizar una función, que use parámetros por valor tipo estructura y que retorne una estructura, para encontrar el menor número impar ingresado, indicando el orden en que ingresó (utilizar un atributo adicional en la estructura para registrar un valor que indique si se ingresó o no al menos un número impar). Presentar por pantalla los resultados.

9. Codificar un programa en el que se solicita el ingreso de la fecha de nacimiento de N personas, luego se deberá ingresar la fecha actual, utilizando una estructura para contener el día, mes y año para las fechas. Luego determinar utilizando una función, que use parámetros por valor tipo estructura y que retorne una estructura, los datos de la persona que tenga la mayor edad. Presentar por pantalla los resultados.

10. Ingresar los datos de N productos alimenticios, los campos de la estructura a usar deben almacenar los siguientes datos: inicial de la marca, precio y fecha de vencimiento (día, mes y año). Luego presentar por pantalla los datos del producto alimenticio que posea la mayor fecha de vencimiento, utilizando una función que use parámetros por valor tipo estructura y que retorne una estructura.

Con funciones con variables simples, referencia, static y recursión

Codificar los siguientes problemas en lenguaje C, empleando ***funciones que retornen o no valor y que utilicen uno o más parámetros por referencia (y paso de parámetros por valor si fuese necesario)***

11. Ingresar dos números complejos (ingresando parte real y parte imaginaria utilizando estructura), luego presentar por pantalla los números complejos ingresados, la suma y diferencia de dichos números. Debe utilizar funciones para sumar los números complejos y para efectuar la diferencia, utilizando estructuras (una o más, en caso de ser necesario) como parámetros por referencia (y por valor si fuese necesario).

12. Ingresar un número real positivo B y un número entero no negativo E (realizar los controles de datos mediante funciones), encontrar el valor de B elevado a E (mediante una función que emplee parámetro por referencia y por valor en caso necesario), utilizando únicamente operaciones de producto. Presentar por pantalla el resultado.

13. Ingresar un número no negativo (realizar el control de datos de entrada mediante una función), presentar por pantalla el número ingresado y el valor del factorial del mismo (se debe utilizar una función para calcular el valor del factorial), utilizar parámetro por referencia (y por valor si fuese necesario).

14. Realizar un programa que permita ingresar dos números enteros positivos N1 y N2 (controlar con una función), luego mostrar por pantalla el resultado de la suma de los números primos comprendidos entre ellos (utilizar una función para calcular la suma, además dicha función debe invocar a otra función para determinar si un número es o no primo), utilizar parámetro por

referencia (y por valor si fuese necesario).

15. Resolver utilizando una o más funciones que retornan valor, no utilicen parámetros y haga uso de una o más variables locales static y auto si fuera necesario. Presentar por pantalla, las N (con N<=27) primeras letras del alfabeto en mayúsculas.

16. Para resolver con funciones recursivas: Ingresar un número entero positivo n, luego imprimir los números del n al 1, utilizando una función recursiva.

Programas que deben utilizar funciones con estructuras y parámetros por referencia

17. Codificar un programa que permita el ingreso de un número entero positivo *numerador* y otro número entero positivo *denominador* (realizar el control de datos de entrada) luego mediante una función que reciba los datos de entrada, calcular el cociente y el resto de la división entera entre el numerador y el denominador utilizando el método de restas sucesivas. La función debe utilizar parámetro por referencia tipo estructura para contener el resultado del resto y del cociente. Presentar por pantalla los resultados.

18. Codificar un programa en el que se solicita el ingreso de dos números complejos (ingresando parte real y parte imaginaria de la forma binómica en una variable tipo estructura para cada número complejo), luego utilizar una función para calcular el producto de los mismos, además debe utilizar parámetro por referencia de tipo estructura para contener el resultado. Presentar por pantalla los números ingresados y el resultado.

19. Se necesita conocer la edad de una persona en años, meses y días, para ello se cuenta con la fecha de nacimiento y con la fecha actual. Realizar un programa que utilice una función para determinar la edad, que use variables tipo estructura para contener los datos de año, mes y día, además deberá utilizar parámetro por referencia tipo estructura. Presentar por pantalla los resultados.

20. Realizar un programa que determine los años, meses y días comprendidos entre la fecha actual y la fecha de vencimiento de un producto alimenticio. Utilizar una función para realizar el cálculo y que use variables tipo estructura que puedan contener los datos de años, meses y días. Presentar por pantalla los resultados o un mensaje en caso de estar vencido. Utilizar parámetros por referencia.

21. En una una carrera, en donde compiten N automóviles, se registra una letra que identifica a cada automóvil y el tiempo que demora en llegar a la meta. Utilizar una función para determinar cuál es la letra que corresponde al automóvil ganador y el tiempo utilizado, emplear variables tipo estructura para contener los datos necesarios y utilizar parámetros por referencia. Presentar por

pantalla los resultados.

Problemas de Profundización, funciones con variables simples

Codificar los siguientes problemas en lenguaje C, empleando ***funciones que retornen o no valor y que usen paso de parámetros por valor***

1. Ingresar dos números complejos (ingresando parte real y parte imaginaria de la forma binómica en variables separadas), luego presentar por pantalla los números ingresados, la suma y la diferencia de dichos números.
2. Ingresar un número entero positivo (controlar con una función), determinar (mediante otra función) si es par o impar utilizando solamente operaciones de sumas sucesivas o restas sucesivas. Presentar por pantalla el mensaje correspondiente.

Codificar los siguientes problemas en lenguaje C, empleando ***funciones que retornen o no valor y que utilicen uno o más parámetros por referencia (y paso de parámetros por valor si fuese necesario)***

3. Realizar un programa que permita ingresar dos números enteros positivos N y B, donde B debe ser menor que 11 y mayor que 1 (realizar el control de datos de entrada mediante funciones), luego presentar por pantalla el número N expresado en la base B (utilizar una función para realizar la conversión).
4. Presentar por pantalla todos los números enteros positivos de cuatro dígitos, que cumplan la condición de que el primer dígito sea igual al último y el segundo dígito sea igual al penúltimo (utilizando una o más funciones).
5. Presentar por pantalla todos los números que tengan cuatro dígitos y que pertenezcan a la serie de Fibonacci. Utilizar una función para determinar la condición solicitada.
6. Presentar por pantalla todos los números de cuatro dígitos que cumplan con la condición de que la suma del cuadrado de los dígitos que se encuentren en posición par sea igual a la suma del cuadrado de los dígitos que se encuentren en posición impar. Ejemplo: Uno de los números que cumplen con dicha condición es: 1784; dado que 64+1=16+49. Utilizar una función para determinar la condición solicitada.
7. Presentar por pantalla los primeros números triangulares menores que N. Utilizar una función para el control de N y otra función para encontrar los números triangulares.

Resolver utilizando funciones que utilicen parámetros por valor, que retornen valor y que hagan uso de variables static (y auto si fuese necesario).

8. Realizar un programa que permita ingresar N números enteros, luego presentar por pantalla la cantidad de series de números impares ingresados. Ejemplo:

4, 9, 2, 5, 3, 2, 15, 3, 7, 9

1era serie 2da serie Resultado: 2 series

Observación: Se considera una serie a partir del segundo elemento que verifica la condición dada.

9. Ingresar N números enteros, luego presentar por pantalla cuantas series de números estrictamente crecientes aparecen. Ejemplo

5, 1, 9, 7, 6, 0, 1, 9, 3, 2, 4, 7, 9

1era serie 2da serie 3era serie

Resultado: 3 series

Resolver utilizando una o más funciones que retornen valor, no utilicen parámetros y haga uso de variables locales static y auto si fuera necesario.

10. Un número triangular es un número que puede recomponerse en la forma de un triángulo equilátero (por convención, el primer número triangular es el 1). Es decir, los primeros números triangulares son 1, 3, 6, 10, etc. Elabore un programa que muestre en pantalla los N primeros números triangulares.
11. Presentar por pantalla los N primeros números de la serie de Fibonacci.

Resolver con funciones recursivas

12. Escribir un programa que permita ingresar dos números enteros positivos y que muestre la suma de dichos números utilizando el método recursivo.
13. Ingresar un número entero no negativo n. Presentar por pantalla el enésimo número de la serie de Fibonacci utilizando una función recursiva.

Resolver utilizando funciones que usen parámetros por valor

14. Presentar por pantalla la sumatoria de los primeros números triangulares menores que un valor entero positivo N ingresado (controlar con una función). Utilizar una función para encontrar los números triangulares.
15. Leer un número entero positivo de dos o más dígitos, deberá hacer el control de datos de entrada empleando una función. Presentar por pantalla al número ingresado y su invertido, para hacer esta tarea, deberá utilizar otra función. Ejemplo: se ingresa 8253 -> Muestra 3528
16. Presentar por pantalla los primeros números primos menores que N. Utilizar una función para controlar que N sea positivo y otra función para determinar si un número es o no primo.
17. Presentar por pantalla todos los números primos que posean cuatro dígitos (utilizar una función para determinar si el número es primo).
18. Ingresar un número positivo en formato binario (controlar con una función), luego encontrar

(mediante otra función) el complemento a la base de dicho número. Presentar por pantalla el número ingresado y su correspondiente complemento a la base.

19. El valor de e^x se puede aproximar mediante la siguiente sumatoria: $e^x = 1 + x + x^2 / 2! + x^3 / 3! + ... + x^n / n!$. Escribir un programa que permita ingresar un valor para x y un entero para n. Presentar por pantalla el valor de la suma resultante. Utilizar funciones para el control de los datos de entrada y para realizar el cálculo de la potencia y del factorial.

Resolver utilizando funciones que usen parámetros por referencia y parámetros por valor si fuese necesario.

20. Ingresar N números, (realizar el control de que N sea positivo con una función), luego presentar por pantalla la cantidad de series estrictamente crecientes (utilizando una función).
21. Ingresar tres caracteres alfabeticos (realizar el control de datos de entrada), luego ordenar (mediante una función) dichos caracteres en forma ascendente. Presentar por pantalla a los caracteres ingresados y ordenados.
22. Ingresar dos números N y D enteros positivos (realizar el control con una función), luego calcular (con otra función) el valor del cociente y resto entre N y D utilizando el método de restas sucesivas. Presentar por pantalla el resultado.
23. Ingresar N números (realizar el control de que N sea positivo con una función), luego presentar por pantalla el mayor número ingresado que pertenezca a la serie de Fibonacci indicando el orden en que ingresó respectivamente o un mensaje en caso de que ningún número pertenezca a la serie (utilizar funciones).
24. Realizar un programa que permita ingresar N números enteros positivos, luego determinar el valor promedio (o media aritmética) de los números ingresados que sean primos (utilizar una función para determinar si un número es primo y otra función para determinar el promedio). Mostrar por pantalla el resultado o un mensaje en caso de que no se hayan ingresado números primos. Recordar que el promedio no es un valor entero.
25. Ingresar N números (realizar el control de que N sea positivo con una función), luego presentar por pantalla el menor valor y el mayor valor de los números ingresados, indicando el orden en que ingresaron respectivamente (utilizar funciones). En caso de que todos los números sean iguales, mostrar el mensaje correspondiente.

Ingresar N números enteros positivos (realizar el control de N con una función) y presentar por pantalla cuantas series crecientes de números primos aparecen (utilizando una función para determinar las series y otra función para determinar si un número es primo).

Ejemplo:

4 5, 7, 13, 6, 2, 3, 11, 10, 5, 5, 3, 8, 2, 3, 11

, 1º 2º 3º Resultado 3 Series.

Observación: Se considera una serie a partir del segundo elemento que verifica la condición dada.

Funciones con estructuras

26. Implementar un programa que permita ingresar la fecha de vencimiento (año, mes y día) de N medicamentos y la fecha actual, determinar la cantidad de medicamentos que se encuentren vencidos. Utilizar una función que realice los cálculos y use variables tipo estructura que

puedan contener los datos de tipo fecha, presentar por pantalla el resultado. Utilizar parámetros por valor y por referencia.

27. De un grupo de N personas, se registra el peso, altura y edad. Realizar un programa que permita calcular el peso promedio, altura promedio y la edad promedio, utilizando una función que use variables tipo estructura para contener los datos o los resultados, presentar por pantalla los resultados. Utilizar parámetros por valor y por referencia.
28. Ingresar N números enteros, luego presentar por pantalla el menor y el mayor valor de los números ingresados, indicando el orden en que ingresaron respectivamente utilizando funciones que usen estructuras y parámetros por valor y por referencia.