Universidad Autónoma del Estado de México
UAEM Texcoco

Carrera:

Ing. Computación (ICO)

Profesor:

M. en C. Hernández Santiago José

Alumno:

Sánchez Martínez Ulises

Grupo: Vespertino Grado: 03

Unidad de Aprendizaje:

Lenguaje Ensamblador

Tarea:

Conversiones de base A a base B

1.- Proporcione el algoritmo para convertir un número de base A a base Decimal, parte entera y fraccionaria del número.

Como apoyo trabajaremos con los siguientes valores, fáciles de explicar pero que de la misma forma trabajaran de manera correcta con números mas grandes.

Base de Origen (Base A)	Número expresado en su base de Origen	Número expresado en base Decimal
2	101.11	5.75
8	30.5	24.625
16	1F.A	31.625

El primer paso es descomponer el número y representarlo en una cadena respetando su valor posicional que va del punto decimal a la izquierda para el caso de los enteros y a la derecha para los valores decimales o flotantes y multiplicarlos por la base de origen elevado de acuerdo a la notación científica y en el caso de los valores hexadecimales simplemente tomamos en cuenta la equivalencia de la letra en valor decimal y hacemos lo mismo.

Ejemplo:

Base Origen 2.

1er Paso: **1** x (2^2) + **0** x (2^1) + **1** x (2^0) + **1** x (2^{-1}) + **1** (2^{-2}) = ¿?
2º Paso: **1** x 4 + **0** x 2 + **1** x 1 + **1** x 0.5 + **1** x 0.250 = 5.75

Base Origen 8.
1er Paso: **3** x (8^1) + **0** x (8^0) + **5** x (8^{-1}) = ¿?
2º Paso: **3** x 8 + **0** x 1 + **5** x 0.125 = 24.625

Base 16.
1er Paso: **1** x (16^1) + **15** x (16^0) + **10** x (16^{-1}) = ¿? (F equivale a un 15 y A, a 10 en decimal).
2º Paso: **1** x 16 + **15** x 1 + **10** x 0.0625 = 31.625

2.- Proporcione el algoritmo para convertir un número de base decimal a base B, parte entera y fraccionaria del número.

Teniendo en cuenta los valores anteriores es hora de regresarlos a su base de origen es decir obtener los siguientes resultados:

Número expresado en base Decimal	Base destino	Número expresado en la base destino (base B)
5.75	2	101.11
31.625	16	1F.A

Primeramente debemos tomar en cuenta que los valores que conformaran nuestra parte entera serán los residuos en caso de que el numero (solo parte entera) sea divisible entre la base destino, y el dividendo en caso de que no sea divisible entre la base destino, teniendo en cuenta que el coeficiente sera nuevamente dividido entre en divisor hasta lograr que este sea menor que la base destino es decir:

Para convertir 5.75 de su base origen que es Decimal a base B en este caso base 2 hacemos lo siguiente:

1er Paso dividir el número en:

Parte entera: 5
Parte fraccionaria: 0.75

2º Paso comenzar a dividir el numero y luego dividir el coeficiente sucesivamente entre la base destino teniendo en cuenta que los valores que nos interesan son los residuos y en caso de que el dividendo sea menor a la base destino, optaremos por interesarnos en el dividendo es decir:

2 = Base destino.
5 = Parte entera.
0.75 = Parte fraccionaria.

Mientras (Parte entera > Base destino) obtener {
Modulo = Residuo de Parte entera / Base destino
Coeficiente = Parte entera / Base destino
Reemplazar (Parte entera = Coeficiente) }

mientras (5>=2) entonces {
__1__ = modulo de 5 / 2
2 = 5 / 2
5 ahora es igual a 2 }

mientras (2>=2) entonces {
__0__ = modulo de 2 / 2
1 = 2 / 2
2 ahora es igual a 1 }

mientras (__1__ >=2) entonces[...]
aquí ya no cumple la condición por tanto ahora tomamos en cuenta nuestro dividendo

Y los ordenamos de forma contraria a la cual fueron obtenidos en este caso parece quedar igual al orden obtenido pero no lo es, finalmente la parte entera queda así: 101

Para sacar la parte fraccionaria tenemos en cuenta que en este caso nos interesan valores enteros resultado de la multiplicación de la parte fraccionaria con la base Destino hasta que la parte flotante de la multiplicación sea igual a 0.0.
Es decir:

Mientras (Parte flotante sea diferente a 0.0) obtén {
Multiplicación = Parte flotante x Base destino
Parte flotante = Multiplicación – Parte entera de multiplicación
}

Mientras (Parte fraccionaria != 0.0) {
Multiplicación = **0.75** *x 2*
Parte entera = **1**
Parte fraccionaria = 0.5
}

Mientras (Parte fraccionaria != 0.0) {
Multiplicación = **0.5** *x 2*
Parte entera = **1**
Parte fraccionaria = 0.0

La parte fraccionaria se ordenan de acuerdo a su aparición por lo tanto quedaría así .11
Quedando como resultado: **101.11**

Ahora para el valor 31.625 realizamos absolutamente lo mismo pero cuando los residuos de las divisiones sean mayores o iguales a 10 debemos remplazar por la letra correspondiente por ejemplo si el residuo es 10 reemplazamos por A, si el residuo es 11 reemplazamos por B… hasta 15 que será reemplazado por F.

16 = Base destino.
31 = Parte entera.
0.625 = Parte fraccionaria.

Mientras (Parte entera > Base destino) obtener {
Modulo = Residuo de Parte entera / Base destino
Coeficiente = Parte entera / Base destino
Reemplazar (Parte entera = Coeficiente) }

mientras (31>=16) entonces {
15 *= modulo de 31 / 16*
1 = 31 / 16
31 ahora es igual a 1 }

mientras (**1** *>=16) entonces[…]*
aquí ya no cumple la condición por tanto ahora tomamos en cuenta nuestro dividendo

Y los ordenamos de forma contraria a la cual fueron obtenidos, finalmente la parte entera queda así:
15 es igual a “F” debido a su equivalencia en hexadecimal y 1 es igual a “1” debido a su equivalencia en hexadecimal.
Resultado de parte entera: 1F

Para sacar la parte fraccionaria tenemos en cuenta que en este caso nos interesan valores enteros resultado de la multiplicación de la parte fraccionaria con la base Destino hasta que la parte flotante de la multiplicación sea igual a 0.0.
Es decir:

Mientras (Parte flotante sea diferente a 0.0) obtén {
Multiplicación = Parte flotante x Base destino
Parte flotante = Multiplicación – Parte entera de multiplicación
}

Mientras (Parte fraccionaria != 0.0*) {*
Multiplicación = **0.625** *x 16*
Parte entera = **10**
Parte fraccionaria = **0.0**
}

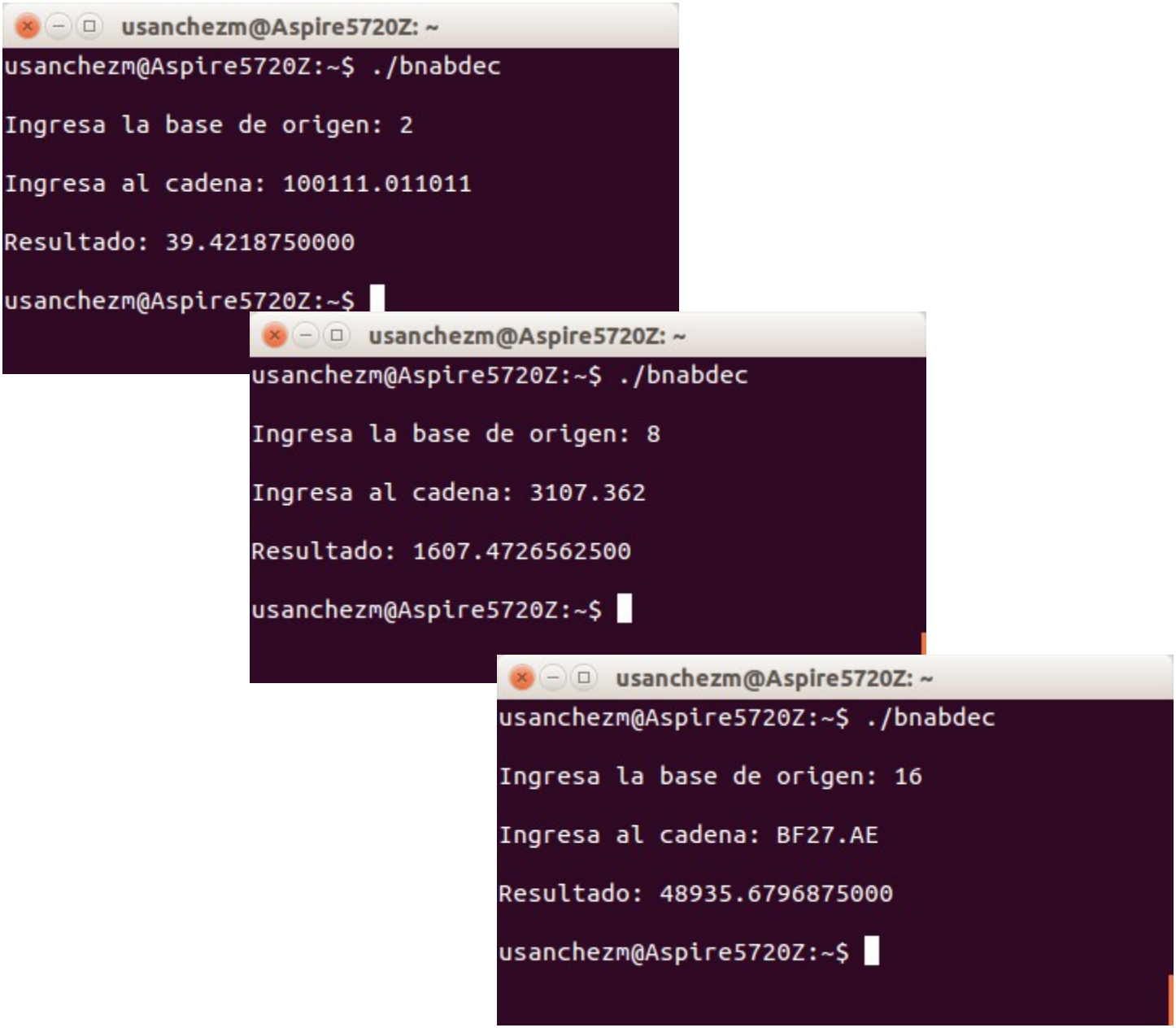
Aqui solo obtuvimos 10 como parte fraccionaria debido a que la multiplicación no dio decimales mayores a 0.0 sino iguales a 0.0 y por lo tanto cumplió con nuestro estado de parada, y como sabemos 10 es igual a "A" debido a su equivalencia en base hexadecimal.
Resultado parte fraccionaria: A

Quedando como resultado: **1F.A**

3.- Programa 1. Realice un programa, en el lenguaje que desee, capaz de convertir un número de alguna base A hacia una base decimal, usando el algoritmo 1. (imprima solo la ejecución)

4.- Aplique la siguiente prueba al Programa 1 y verifique que los datos sean correctos.

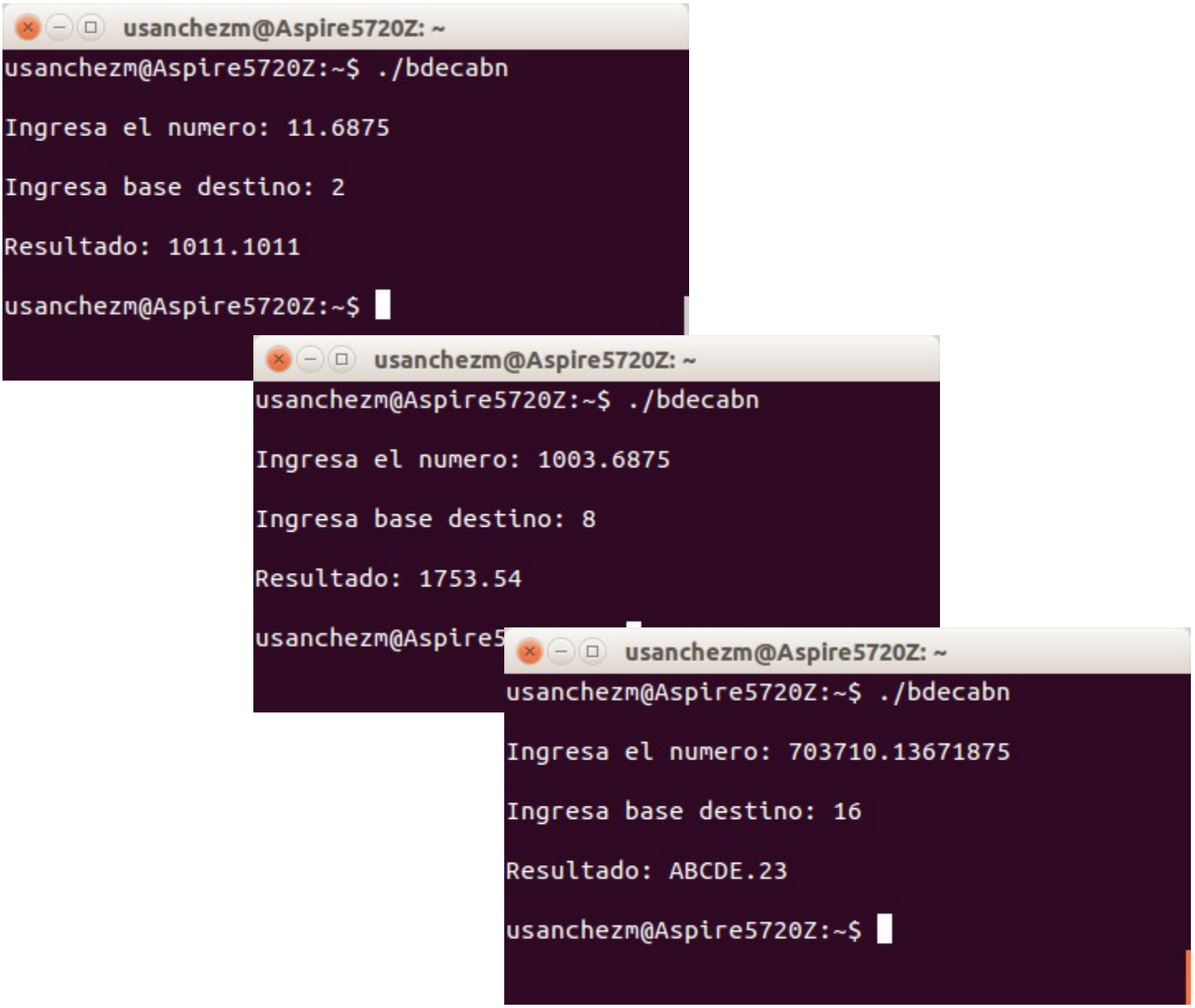
Base de Origen (Base A)	Número en Base A	Base destino 10	Acertó / Fallo
2	100 111 . 011 011	39.421875	1
8	3107.362	1607.47265 625	1
16	BF27.AE	48 935.6796875	1



5.- Programa 2. Realice un programa, en el lenguaje que desee, capaz de convertir un número de base decimal hacia una base B, usando el algoritmo 2. (imprima solo la ejecución)

6.- Aplique la siguiente prueba al Programa 2 y verifique que los datos sean correctos.

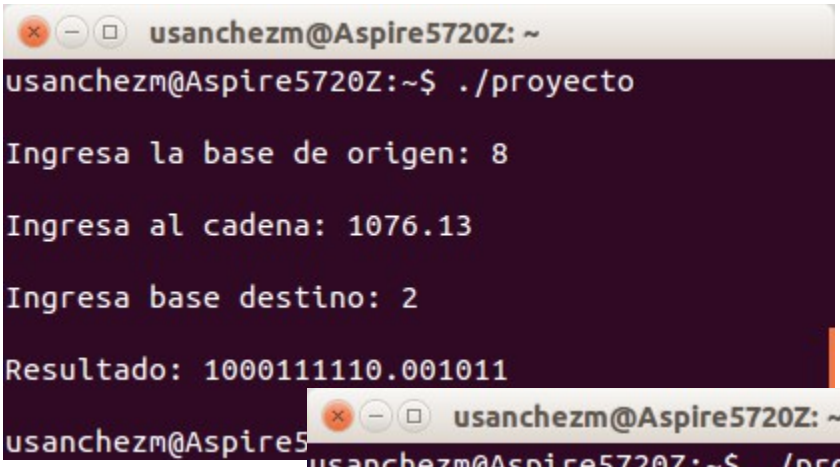
Numero decimal (Base de Origen 10)	Base Destino	Número en Base B	Acertó / Fallo
11.6875	2	1011.1011	1
1003.6875	8	1753.54	1
703 710.13671 875	16	ABCDE.23	1

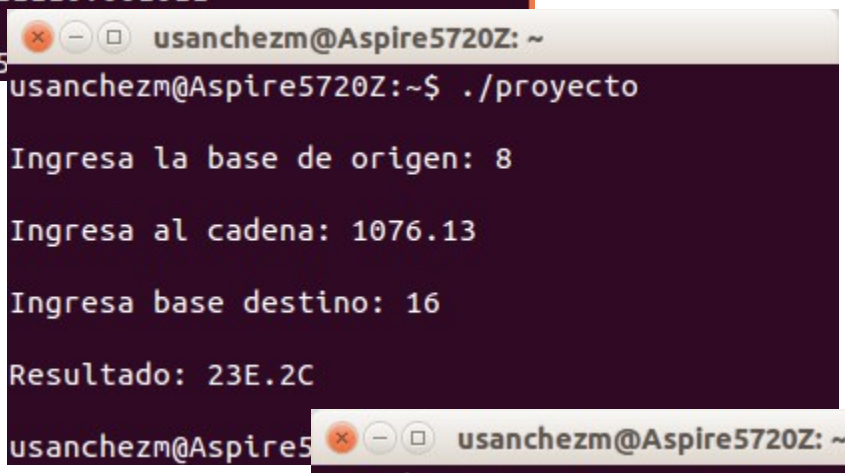


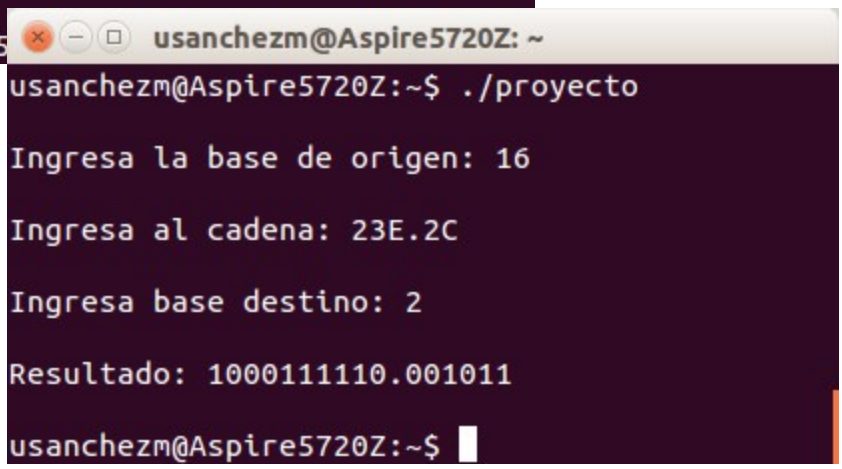
7.- Programa 3. Utilice los dos programas anteriores para elaborar un programa capaz de convertir un número de base A hacia base B.

8.- Aplique la siguiente prueba al Programa 3 y verifique que los datos sean correctos

Base de Origen	Número en Base A	Base de Destino	Número en Base B	Acertó / Fallo
2	1 000 111 110.001011	8	1076.13	1
2	1 000 111 110.001011	16	23E.2C	1
8	1076.13	2	1 000 111 110.001011	1
8	1076.13	16	23E.2C	1
16	23E.2C	2	1 000 111 110.001011	1
16	23E.2C	8	1076.13	1


usanchezm@Aspire5720Z: ~
usanchezm@Aspire5720Z:~$ ./proyecto
Ingresa la base de origen: 8
Ingresa al cadena: 1076.13
Ingresa base destino: 2
Resultado: 1000111110.001011
usanchezm@Aspire5


usanchezm@Aspire5720Z: ~
usanchezm@Aspire5720Z:~$ ./proyecto
Ingresa la base de origen: 8
Ingresa al cadena: 1076.13
Ingresa base destino: 16
Resultado: 23E.2C
usanchezm@Aspire5


usanchezm@Aspire5720Z: ~
usanchezm@Aspire5720Z:~$ ./proyecto
Ingresa la base de origen: 16
Ingresa al cadena: 23E.2C
Ingresa base destino: 2
Resultado: 1000111110.001011
usanchezm@Aspire5720Z:~$


usanchezm@Aspire5720Z: ~
usanchezm@Aspire5720Z:~$ ./proyecto
Ingresa la base de origen: 16
Ingresa al cadena: 23E.2C
Ingresa base destino: 8
Resultado: 1076.13
usanchezm@Aspire5720Z:~$

9.- Proporcione el código documentado para el Programa 3.

```
#include <stdio.h>       // librerías para entrada y salida así también como las que permiten usar funciones como pow
#include <stdlib.h>
#include <math.h>
int bnabdec();  //función para convertir de base n a base decimal
int bdecabn();  //función para convertir de base decimal a base n
int nsti[25],bini,pot,multi,updi,c,totali; //declaracion de variables enteras
double multf,updf,totalf,resultado; //declaracion de variables flotantes
char nstc[25],y[]={"."},z[]={"\0"},letra[]={'A','B','C','D','E','F'};      //cadenas de comparación
int ord,i,j,mod,numi,cof,bdes,mi;
float numf,mult,mf;
char orden[25],val;
int main(){            //funcion principal
printf("\nIngresa la base de origen: ");  //pido la base de origen
scanf("%d",&bini);
printf("\nIngresa al cadena: ");          //pido la cadena a convertir
scanf("%s",&nstc[c]);
bnabdec();                                //mando a llamar mi función de cualquier base a decimal
printf("\nIngresa base destino: ");       //una vez obtenido valor decimal, pido base destino
scanf("%d",&bdes);
bdecabn();                    //mando a llamar función para convertir de base decimal a base n
}
int bnabdec(){                //comienza función base n a base decimal
c=0;                          //inicializo variable
do{
c++;                          //incrementa asta que encuentre el . Decimal asi obtengo tamaño de array
}while(nstc[c]!=y[0]);
pot=c-1; //ajuste variable pot = c-1, para que inicie de acuerdo a la notacion científica
totali=0; //solo inicializo con valor 0 este contendra el resultado de la parte entera
c=0;      //inicializo en 0 c es la posicion de la cadena
do{             //hago ajuste si es base 16 convierto letras a número
if(bini==16&&nstc[c]==letra[0]||nstc[c]==letra[1]||nstc[c]==letra[2]||nstc[c]==letra[3]||
   nstc[c]==letra[4]||nstc[c]==letra[5]){
nsti[c]=nstc[c]-55;
   } else {
nsti[c]=nstc[c]-48;
```

```
                }
multi=pow(bini,pot);
totali=(nsti[c]*multi)+totali; //voy actualizando la suma total
c++;            //c va incrementando porque recorre la cadena
pot--;          //potencia va decrementando ya que inicio desde la posicion 0 primer valor del array
}while(nstc[c]!=y[0]); // se detiene hasta que encuentra el "."
pot=-1;         //potencia se inicializa en -1
totalf=0;       //inicializo variable en 0 aqui acumulo resultado final
c=c+1;          //c va incrementando
do{
if(bini==16&&nstc[c]==letra[0]||nstc[c]==letra[1]||nstc[c]==letra[2]||nstc[c]==letra[3]||
   nstc[c]==letra[4]||nstc[c]==letra[5]){
nsti[c]=nstc[c]-55;                 //convierto de variable char a variable entera y ajusto valor
   } else {
nsti[c]=nstc[c]-48;          }      //convierto de variable char a variable entera y ajusto valor
multf=pow(bini,pot);
totalf=(nsti[c]*multf)+totalf;      //realizo operaciones correspondientes
c++;                                //c incrementando para seguir recorriendo el array
pot--;                              //potencia se va ajustando de acuerdo a la notación científica
}while(nstc[c]!=z[0]);              //termina el proceso hasta encontrar fin de cadena
resultado=totali+totalf;            //guardo resultado en variable resultado que es flotante
}
int bdecabn(){                      // inicia funcion base decimal a base n
numi=totali;                        // heredo total de parte entera a variable numi
numf=totalf;                        // heredo total de parte flotante a variable numf
i=0;                                //inicio variable en 0
while(numi>=bdes){                  //inicio operaciones correspondientes mientras numero > a base (B)
mod=numi%bdes;
if(bdes==16&&mod<10){               //ajusto variables en caso de ser <10 + 48 para obtener valor real
mod=mod+48;                         //en caso de que base destino sea 16
}else
if(bdes==16&&mod>=10){              //ajusto variables en caso de ser >10 + 55 para obtener Letra
mod=mod+55;
}else {
mod=mod+48;                         //en caso de no ser base final 16 hace ajustes normales
}
```

```
orden[i]=mod;                //array para guardar el modulo y luego escribirlo ordenadamente
cof=numi/bdes;
numi=cof;
i++;                         //i es la posicion del array orden
}
if(bdes==16&&numi<10){       //hago lo mismo para ajustar variables para empezar con parte flotante
numi=numi+48;
}else
if(bdes==16&&numi>=10){
numi=numi+55;
}else
{
numi=numi+48;
}
orden[i]=numi;
printf("\nResultado: ");
for(j=i;j>=0;j--){           //con mi for escribo valores guardados en array llamado orden
printf("%c",orden[j]);
          }
printf(".");                 //imprimo un punto
do{                          //inicio operaciones correspondientes
mult=numf*bdes;
mi=mult;
mf=mult-mi;
numf=mf;                     //actualizo numero flotante por parte flotante de la multiplicación
val= int (mi);               //convierto parte entera de multiplicación a tipo char

   if(bdes==16&&mi<10){      //inicio ajuste de valores
          val=val+48;
                } else
   if(bdes==16&&mi>=10){
          val=val+55;
                } else{
val=val+48;
                }
printf("%c",val);            //voy escribiendo el resultado flotante ya ajustado
```

```
    }while(numf!=0.0);        //hasta que mi parte flotante de la multiplicación sea igual a 0.0
printf("\n\n");               //detengo la operación y puse 2 saltos de linea
}
```

10.- Proporcione una conclusión por cada integrante del equipo, respecto a los algoritmos y su programación.

Mi conclusión es que al intentar programar uno debe partir desde un análisis a lápiz y papel para ver el comportamiento de las operaciones, los valores que me interesan y los cambios de valores con los que se repiten las operaciones, las condiciones y valores de parada.

Un detalle que hace mas complejo el algoritmo es cuando hacemos uso de letras, debemos de hacer un cambio de caracter a número, y ay muchos pasos a seguir, desde hacer uso de un if para cuando el número es un 15 imprimir una letra F, o según el caso ó hacer uso de los valores ASCII, un dato curioso es que el numero 1 en tipo char al convertirlo a tipo entero daba como resultado 49 y lo que hacia era restarle 48, eso me permitía obtener el valor original 1, e incluso las letras como la A valía 65 a esas le reste 55 para obtener el valor 10 en su conversión a entero, y luego hacia el proceso inverso según la parte de mi algoritmo lo requiriera.

Como dice un dicho, ay muchos caminos para llegar a Roma, inicialmente me quedaban muchas lineas de código y la versión final quedo más reducida que mis versiones iniciales, cada ves que me sentaba a programar se me ocurrían varias ideas hasta que por fin fui puliendo mis algoritmos y finalmente me quedo mi código, resolviendo con mucha efectividad y eficiencia las pruebas a las que fue sometido para validar su funcionamiento.

En fin, programar es un arte, que puede expresar nuestro modo de pensar y de resolver las cosas. Y supongo que si siguiera trabajando con mi código cada ves obtendría nuevas mejoras.

Nota:

Debe llevar portada con logotipo institucional. ✓

Datos de la asignatura. ✓

Nombres ordenados alfabéticamente. ✓

Entrega una semana antes del examen correspondiente al primer parcial. ✓

El código debe llevar tamaño de fuente de 9. ✓