TALLER

KELIN ALEJANDRA ARBOLEDA DAVILA

CORPORACIÓN DE ESTUDIOS

TECNOLÓGICOS DEL NORTE DEL VALLE

TÉCNICO PROFESIONAL EN PROGRAMACIÓN

DE APLICACIONES INFORMÁTICAS

CARTAGO 2017

**1. Cada una de las siguientes expresiones escritas en infijo convertirlas a posfijo:**

a)  (x+y)\*(z^(s+t)+u)-v

(x y +) \* (z ^ (s t +) + u) - v

(x y +) \* (z s t + ^) + u) - v

((x y +) \* (z s t + ^ u +)) - v

x y + z s t + ^ u + \* - v

**x y + z s t + ^ u + \* v –**

b)  a+((x+y)\*(w+v))^(i+j)

a + ((x y +) \* (w v +)) ^ (i + j)

a + ((x y + w v + \*) ^ (i + j))

a + (x y + w v + \* i j + ^)

**a x y + w v + \* i j + ^ +**

c)  (a+b)\*(c+d)

(a b +) \* (c d +)

**a b + c d \***

d)  a+b-c

**ab+c-**

e)  (a+b)\*(c-d)\*e\*f

((a b +) \* (c d -) \* e \* f)

(((a b + c d - \*) \* e) \* f)

((a b + c d - \* e \*) \* f)

**a b + c d - \* e \* f \***

f)  (a+b)\*(c/(d-e)+f)-g

(a b +) \* ((c / (d e -)) + f) - g

(a b +) \* ((c d e - /) + f) - g

((a b +) \* (c d e - / f +)) - g

((a b + c d e - f + \*) - g)

**a b + c d e - / f + \* g -**

g)  (a^b^c)

(a b ^) ^ c

**a b ^ c ^**

h)  a^b\*c-d+e/f/(g+h)

((((a ^ b) \* c) - d) + ((e / f) / (g h +)))

((((a b ^) \* c) - d) + ((e f /) / (g h +)))

((((a b ^ c \*) - d) + ((e f /) / (g h +)))

(((a b ^ c \*) - d) + (e f / g h + /))

((a b ^ c \* d -) + (e f / g h + /))

**a b ^ c \* d - e f / g h + / +**

 i)   a+(((b-c)\*(d-e)+f)/g)^(h-j)

a + (((b c -) \* (d e -) + f) / g) ^ (h j -)

a + ((b c - d e - \* f) / g) ^ (h j -)

a + ((b c - d e - \* f g /) ^ (h j -))

(a + (b c - d e - \* f g /)) ^ (h j -)

((a b c - d e - \* f g / +) ^ (h j -))

**a b c - d e - \* f g / + h j - ^**

**2. Convertir las expresiones del ejercicio anterior a prefijo.**

1. (x + y) \* (z ^ (s + t) + u) - v

(x + y) \* (((z ^ (s + t)) + u) - v

(+ x y) \* (((^ z + s t) + u) - v

(+ x y) \* (+ ^ z + s t u) - v

(+ x y) \* (- + ^ z + s t u v)

**\* + x y - + ^ z + + s t u v**

b. a + ((x + y) \* (w + v)) ^ (i + j)

(a + ((x + y) \* (w + v))) ^ (i + j)

a + ((+ x y) \* (+ w v)) ^ (+ i j)

(a + (\* + x y + w v)) ^ (+ i j)

((+ a \* + x y + w v) ^ (+ i j))

**^ + a \* + x y + w v + i j**

1. a + b) \* (c + d)

(+ a b) \* (+ c d)

**\* + a b + c d**

d. ((a + b) - c)

((+ a b) - c)

**- + a b c**

e. (a + b) \* (c - d) \* e \* f

((+ a b) \* (- c d)) \* e \* f

((\* + a b - c d) \* e) \* f

(\* \* + a b - c d e) \* f

**\* \* \* + a b - c d e f**

f. (a + b) \* (c / (d - e) + f) - g

(+ a b) \* ((c / (- d e)) + f) - g

(+ a b) \* ((/ c - d e) + f) - g

((+ a b) \* (+ / c - d e f)) - g

((\* + a b + / c - d e f) - g)

**- \* + a b + / c - d e f g**

g. ((a ^ b) ^ c)

((^ a b) ^ c)

**^ ^ a b c**

h. a ^ b \* c - d + e / f / (g + h)

(((^ a b) \* c) - d) + (e / (f / (g + h)))

((\* ^ a b c) - d) + (e / (f / (g + h)))

(- \* ^ a b c d) + (e / (f / (g + h)))

(- \* ^ a b c d) + (e / (f / (+ g h)))

(- \* ^ a b c d) + (e / (/ f + g h))

(- \* ^ a b c d) + (/ / e f + g h)

**+ - \* ^ a b c d // e f + g h**

i. a + (((b - c) \* (d - e) + f) / g) ^ (h - j)

a + ((((- b c) \* (- d e)) + f) / g) ^ (- h j)

a + (((\* - b c - d e) + f) / g) ^ (- h j)

a + ((+ \* - b c - d e f) / g) ^ (- h j)

(a + (/ + \* - b c - d e f g)) ^ (- h j)

(+ a / + \* - b c - d e f g) ^ (- h j)

**^ + a / + \* - b c - d e f g - h j**

**3. Convertir las siguientes expresiones de prefijo a infijo.**

1. + - ^ x y z \* i \* \* k l m

**((x ^ y) - z) + (i \* ((k \* l) \* m))**

1. + + l - \* ^ m n o / + p q \* r s t

**(l + ((m ^ n) \* o)) + ((p + q) - (r \* s) / t)**

**4. Convertir las siguientes expresiones en posfijo a infijo.**

a)  p q r s t - + ^ \* u v \* -

**(p \* (q ^ (r + (s + t)))) - (u \* v)**

b) l m - n + o p q - + ^

**((l - m) + n) ^ (o + (p - q))**

/\*

\*Nombre: Taller 3

\*Fecha: 19-10-2017

\*Presentado por: Kelin Alejandra Arboleda Davila.

\*/

#include <stdio.h>

#include <stdlib.h>

//estructura para un siguiente nodo

struct Nodo{

int dato;

struct nodo \*sgte;

};

//prototipo de las funciones

void menu();

void init\_pila();

bool vacia\_pila();

void ins\_pila();

void retirar\_pm();

void concatenar\_2\_listas();

void sumar\_2\_listas();

void dividir\_2\_listas();

void Lista\_potencia\_otra\_lista();

void Copiar\_lista\_en\_otra();

//funcion principal

int main(){

menu();

return 0;

}

void menu(){

int opcion;

printf("\n\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*");

printf("\n\* 1. Init\_pila \*");

printf("\n\* 2. Vacia\_pila \*");

printf("\n\* 3. Ins\_pila \*");

printf("\n\* 4. Retirar\_pm \*");

printf("\n\* 5. Concatenar 2 listas \*");

printf("\n\* 6. Sumar 2 listas \*");

printf("\n\* 7. Dividir 2 listas \*");

printf("\n\* 8. Lista ^ otra lista \*");

printf("\n\* 9. Copiar lista en otra \*");

printf("\n\* 0. Salir \*");

printf("\n\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n");

printf("\nIngrese Un Numero: ");

scanf("%d", &opcion);

system("cls");

switch(opcion){

case 1: init\_pila();

break;

case 2: vacia\_pila();

break;

case 3: ins\_pila();

break;

case 4: retirar\_pm();

break;

case 5: concatenar\_2\_listas();

break;

case 6: sumar\_2\_listas();

break;

case 7: dividir\_2\_listas();

break;

case 8: Lista\_potencia\_otra\_lista();

break;

case 9: Copiar\_lista\_en\_otra();

break;

case 0: exit(0);

break;

default:

printf("Opcion Invalida");

}

}

void init\_pila(){

}

bool vacia\_pila(){

if(cima==NULL){

return true;

}else{

return false;

}

}

void ins\_pila(){

}

void retirar\_pm(){

}

void concatenar\_2\_listas(){

}

void sumar\_2\_listas(){

}

void dividir\_2\_listas(){

}

void Lista\_potencia\_otra\_lista(){

}

void Copiar\_lista\_en\_otra(){

}