

Instituto Politécnico Nacional

Escuela Superior de Cómputo



Estructuras de Datos

Practica 01: Evaluación de expresiones infijas

M. en C. Edgardo Adrián Franco Martínez

<http://www.eafranco.com>

edfrancom@ipn.mx

[@edfrancom](#) [edgardoadrianfrancom](#)





Contenido

- Definición del problema
- Actividades
- Observaciones
- Requerimientos del código en C
- Reporte de práctica
- Rubrica de evaluación del reporte
- Entrega vía Web





Definición del problema

- Con la implementación del TAD Pila en C, (*estática y dinámica*) implementar un programa que valide y evalúe una expresión infija.
 - Ejemplo
 - $A*((B+D)/C)+E^A$
- Considerar la precedencia de operadores siguiente:
 - 1. Términos entre paréntesis ()
 - 2. Potencia ^
 - 3. Multiplicación y división * /
 - 4. Suma y resta + -
- 1. ***Evaluación de paréntesis escritos correctamente***
 - *El programa mostrara el resultado de la revisión de paréntesis*
- 2. ***Conversión a posfijo***
 - *El programa mostrará el resultado de pasar la expresión infija a posfija.*
- 3. ***Evaluación de la expresión posfija***
 - *El programa mostrará el resultado de la evaluación de la expresión, para ello solicitara el valor flotante de cada una de las variables (A-Z) que contiene la expresión.*





Actividades

1. Programar en ANSI C, el algoritmo para validar paréntesis en una expresión aritmética. Emplee el algoritmo que se apoya de una pila para ello.
2. Programar en ANSI C, el algoritmo para pasar una expresión aritmética en posfijo a partir de la expresión infija con la ayuda de una pila.
3. Programar en ANSI C, el algoritmo para evaluar una expresión aritmética posfija mediante la ayuda de una pila.
4. Crear el programa final que realiza todas las actividades anteriores de manera agradable para el usuario.





Observaciones

- Longitud máxima de la expresión aritmética será de 100 caracteres.
- Los nombres de los operandos en la expresión pueden ser representados con las letras A,B,C,D,...Z.
- Los valores numéricos de los operandos se solicitarán cuando se desee ver el resultado de la evaluación y son números reales.
- **Deberá de usar las operaciones del TAD pila especificadas en clase. No deberá de modificarse la implementación de la pila.** *Solo agregar los campos requeridos en la estructura “elemento”.





```
typedef struct elemento
{
    //Variables de la estructura
    "elemento" (El usuario puede modificar)
    char c;
    //***
    //***
    //***
}elemento;
```

RECUERDE QUE SE SE MODIFICA LA ESTRUCTURA ELEMENTO Y SE DESEA
USAR EL CÓDIGO OBJETO COMO LIBRERÍA EN LUGAR DEL .C SE DEBERÁ
GENERAR EL CÓDIGO OBJETO NUEVAMENTE:

```
gcc -c TADPilaDin.c
```

```
gcc -c TADPilaEst.c
```



- El número de integrantes por equipo es de 2 a 3.



- Recordar conclusiones individuales.
- En la revisión del programa deberán de dominar todos los aspectos de su programa, ya que al azar se preguntara sobre distintas partes de su código y solución. *(Se sorteará a la persona que entregará la practica)*






Requerimientos del código en C

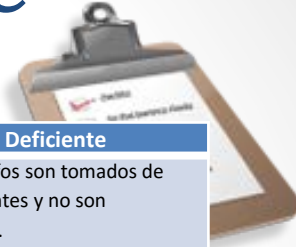
- Documentado (Nombre de los alumnos, versión, explicación del programa)
- El nombre de las variables deberá ser adecuado y entendible (En español)
- La implementación de la pila se maneja en archivos separados
- Las operaciones de la pila (En ingles) según la especificación dada.
- Documentación de funciones y partes importantes de los códigos según el objetivo del programa y la teoría vista en clase.
 - Parámetros que recibe y devuelve
 - Posibles errores o excepciones no soportadas
- Se deberá de probar con ambas implementaciones de la pila (estática y dinámica)
- Instrucciones de compilación y ejecución detalladas.
- Código autodocumentado (Nombres de variables y funciones adecuados y entendibles).



Reporte de practica

- Portada
- Introducción
- Planteamiento del problema
- Diseño y funcionamiento de la solución (Descripción de la abstracción del problema y su solución, apoyándose de diagramas y figuras en un lenguaje claro)
*Descripción del funcionamiento de los algoritmos que se apoyan del TAD pila.
- Implementación de la solución (Según la solución diseñada como se implemento en el lenguaje de programación)
- Funcionamiento (Verificación de la solución, pruebas y resultados de salida
*Pantallazos)
- Errores detectados (Si existe algún error detectado, el cuál no fue posible resolver o se desconoce el motivo y solo ocurre con ciertas condiciones es necesario describirlo)
- Posibles mejoras (Describir posibles disminuciones de código en la implementación o otras posibles soluciones)
- Conclusiones (Por cada integrante del equipo)
- Anexo (Códigos fuente *con colores e instrucciones de compilación)

 Bibliografía (En formato IEEE)



Rubrica de evaluación del reporte

Indicador	Excelente	Muy bien	Bien	Deficiente
Construcción de párrafos	Todos los párrafos incluyen una introducción, explicaciones o detalles y una conclusión	Los párrafos incluyen información relacionada pero no fueron generalmente bien organizados	La estructura del párrafo no estaba clara y las oraciones no estaban generalmente relacionadas	Los párrafos son tomados de otras fuentes y no son originales.
Redacción	No hay errores de gramática, ortografía y puntuación y la redacción es coherentemente	No hay errores de gramática, ortografía y puntuación, pero la redacción presenta incoherencias	Pocos errores de gramática, ortografía y puntuación	Muchos errores de gramática, ortografía y puntuación
Cantidad de información <small>Portada, Introducción, Planteamiento del problema, algoritmos e implementación, actividades y pruebas, errores detectados, posibles mejoras, conclusiones y anexos</small>	Todos los temas son tratados de manera clara y precisa, según lo solicitado.	La mayoría de los temas son tratados de manera clara y precisa	Dos temas no están tratados o están imprecisos y no cumplen lo solicitado.	Tres o más temas no están tratados o están imprecisos y no cumplen lo solicitado.
Calidad de la información	La información está claramente relacionada con el tema principal y proporciona varias ideas secundarias y/o ejemplos	La información da respuestas a las preguntas principales, y solo da algunos detalles y/o ejemplos	La información da respuestas a las preguntas principales, pero no da detalles y/o ejemplos	La información tiene poco o nada que ver con las preguntas planteadas.
Algoritmos	Los algoritmos dan solución apoyándose de pseudocódigo, diagramas y/o figuras en un lenguaje claro.	La mayoría de los algoritmos dan solución apoyándose de pseudocódigo, pero diagramas y/o figuras.	Los algoritmos son mencionados textualmente pero no se describen	Los algoritmos no son expresados en el reporte.
Organización	La información está muy bien organizada con párrafos bien redactados y con subtítulos con estilos adecuados	La información está organizada, pero no se distingue en estilos adecuados	La información está organizada, pero los párrafos no están bien redactados	La información proporcionada no parece estar organizada o es copiada de referencias externas de manera literal



Entrega vía Web



Grupo	Contraseña
1CM12	Estructuras1cm12
1CM13	Estructuras1cm13

- **En un solo archivo comprimido** (ZIP, RAR, TAR, JAR o GZIP)
 - Reporte (DOC, DOCX o PDF)
 - Códigos fuente (.C, .H, etc.)
 - **Código documentado:** Titulo, descripción, fecha, versión, autor.
 - *(Funciones y Algoritmos: ¿Qué hace?, ¿Cómo lo hace?, ¿Qué recibe?, ¿Qué devuelve?, ¿Causa de errores?).*
 - OBSERVACIONES
 - *NO enviar ejecutables o archivos innecesarios, las instrucciones de compilación van en el anexo del reporte. (Yo compilare los fuente)



Fechas de entrega



- **Demostración** *Laboratorio de Programación 1 (1107)*
- 1CM12, 1CM13 “Miércoles 30 de agosto o miércoles 06 de septiembre de 2017”.



Entrega de reporte y código

- En un solo archivo comprimido.



Fecha y hora limite de entrega vía Web

- Miércoles 13 de Septiembre de 2017 a las 23:59:59 hrs.

