|  |  |  |
| --- | --- | --- |
| **CARRERA:**  Ingeniería de Software | **GUÍA**  No. 01 | **TIEMPO ESTIMADO:**  8 horas. |
| **ASIGNATURA:**  Estructura de Datos  NRC: 3251 | **FECHA DE ELABORACIÓN:** 03 de marzo del 2021 **SEMESTRE**: Nov 2020 – Abr 2021 | |
| **TÍTULO:**  Proyecto Segundo Parcial | **DOCENTE:**  Ing. Fernando Solis | |

|  |
| --- |
| 1. **INTRODUCCIÓN** |

1. **Notación prefija y postfija de expresiones aritméticas**

Las operaciones aritméticas escritas en notación infija en muchas ocasiones necesitan usar paréntesis para indicar el orden de evaluación. Las expresiones

Son distintas al no poner paréntesis en la expresión g. Igual ocurre con estas otras:

Existen otras formas de escribir expresiones aritméticas, que se diferencian por la ubicación del operador respecto de los operandos. La notación en la que el operador se coloca delante de los dos operandos, notación prefija, se conoce también como notación polaca por el matemático polaco que la propuso (Joyanes Aguilar, Sanchez García, & Zahonero Martínez, 2007).

1. **Notación polaca**

La propiedad fundamental de la notación polaca es que el orden de ejecución de las operaciones está determinado por las posiciones de los operadores y los operandos en la expresión. No son necesarios los paréntesis al escribir la expresión en notación polaca. La notación postfija o polaca inversa coloca el operador a continuación de sus dos operandos (Guardati, 2007).

Las diferentes formas de escribir una misma expresión algebraica dependen de la ubicación de los operadores respecto a los operandos. Es importante tener en cuenta que tanto en la notación prefija como en la postfija no son necesario los paréntesis para cambiar el orden de evaluación.

1. **Evaluación de una expresión aritmética**

La evaluación de una expresión aritmética escrita de manera habitual, en notación infija, se realiza en dos pasos principales:

* Transformar la expresión de notación infija a postfija.
* Evaluar la expresión en notación postfija.

El orden que fija la estructura pila asegura que el último en entrar es el primero en salir, y de esa forma el algoritmo de transformación a postfija sitúa los operadores después de sus operandos, con la prioridad o precedencia que le corresponde . Una vez que se tiene la expresión en notación postfija, se utiliza otra pila, de elementos numéricos, para guardar los valores de los operandos, y de las operaciones parciales con el fin de obtener el valor numérico de la expresión.

1. **Transformación de una expresión *infija* a *postfija***

Se parte de una expresión en notación infija que tiene operandos, operadores y puede tener paréntesis. La transformación se realiza utilizando una pila para guardar operadores y los paréntesis izquierdos. La expresión aritmética se lee del teclado y se procesa carácter a carácter (Joyanes Aguilar, Sanchez García, & Zahonero Martínez, 2007). Los operandos pasan directamente a formar parte de la expresión en postfija la cual se guarda en un array. Un operador se mete en la pila si se cumple que:

* La pila está vacía
* El operador tiene mayor prioridad que el operador cima de la pila
* El operador tiene igual prioridad que el operador cima de la pila y se trata de la máxima prioridad

|  |
| --- |
| 1. **OBJETIVO** |

Desarrollar la calculadora polaca en lenguaje c++, a través de la utilización de estructuras de datos revisadas en clase para calcular expresiones algebraicas ingresadas por la polaca inversa.

|  |
| --- |
| 1. **DESARROLLO** |

1. **Recursos**
2. Un estudiante por computador.
3. Repositorios de bases digitales, bibliotecas virtuales, y revistas científicas.
4. Material guía que se encuentra compartido en el aula virtual
5. **Desarrollo del taller práctico**
6. Desarrollar la calculadora polaca inversa que reciba como entrada una cadena de caracteres que contenga la notación infija de las operaciones y la convierta en notación posfija para las resolver la expresión ingresada e imprima la respuesta por pantalla usando estructuras de datos lineales como pilas y listas.

|  |
| --- |
| 1. **CONCLUSIONES** |

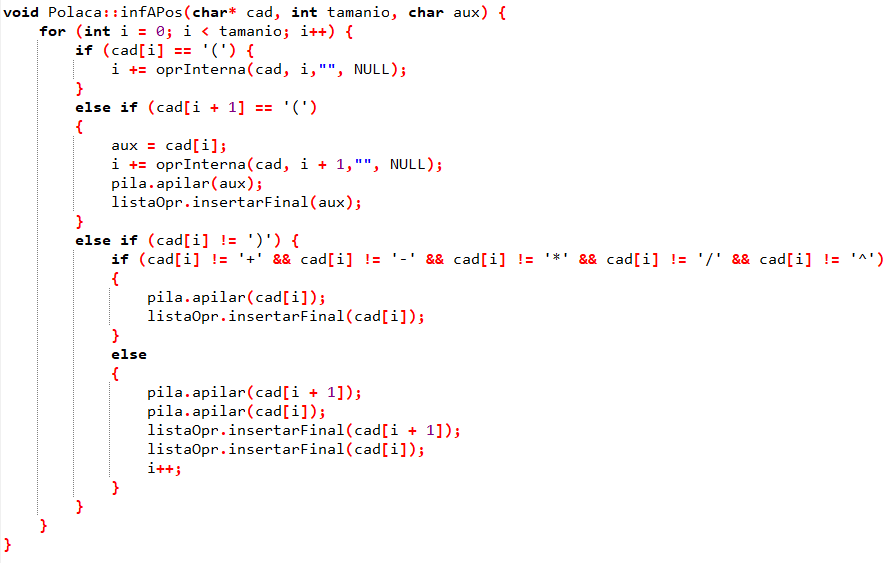
1. Para la implementación de la calculadora polaca se hizo uso de listas dobles y pilas para la conversión de notación infija a posfija, aunque para ciertas expresiones matemáticas que incluyen paréntesis y jerarquía de operadores no se obtuvo el resultado exacto.
2. El mayor problema que se identifico es la correcta conversión de una notación infija a posfija, esto debido a la variedad de casos que se pudo apreciar al pasar de notación infija a posfija, además de la operación para obtener el resultado.
3. La mejor opción para el desarrollo de la calculadora polaca inversa es la implementación de árboles binarios, esto debido a su estructura no lineal, que permite un mejor recorrido de sus nodos, de esta manera se obtendrían las respuestas correctas a expresiones matemáticas más complejas.

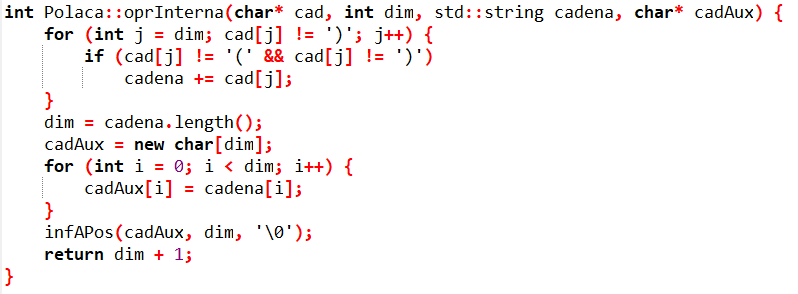
|  |
| --- |
| 1. **BIBLIOGRAFÍA** |

Guardati, S. (2007). *Estructura de Datos Orientada a Objetos.* México: PEARSON Prentice Hall.

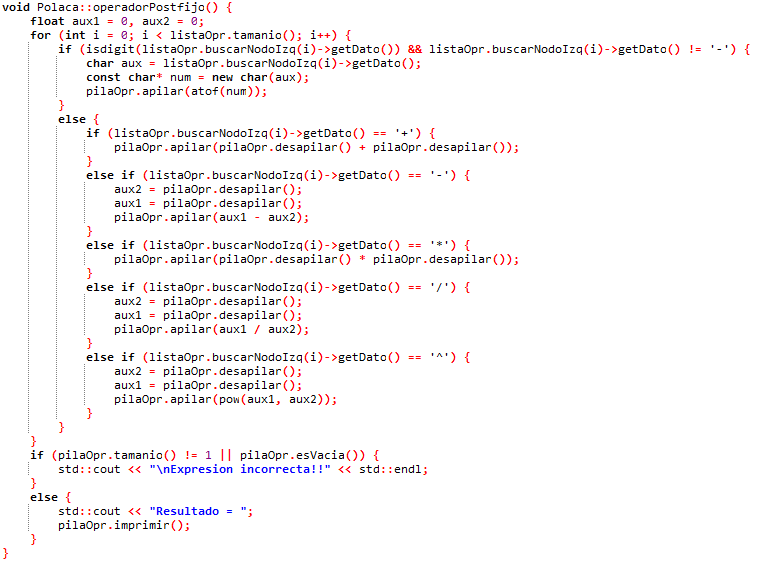
Joyanes Aguilar, L., Sanchez García, L., & Zahonero Martínez, I. (2007). *Estructura de datos en C++.* Madrid: McGRAW-HILL/INTERAMERICANA DE ESPAÑA.

|  |
| --- |
| 1. **ANEXOS** |

1. void infAPos(**char**\*, **int**, **char**) 
2. **int** oprInterna(**char**\*, **int**, std::**string**, **char**\*)



1. **void** operadorPostFijo(**void**)



Text

Description automatically generated

Text

Description automatically generated

Text

Description automatically generated

|  |
| --- |
| 1. **NOMBRES, APELLIDOS Y FIRMA DE LOS ESTUDIANTES PARTICIPANTES** |

Estudiante(s): Chuquimarca Llulluna Kevin Santiago

Iza Quinatoa Marco Alexander

|  |
| --- |
| 1. **FECHA DE ENTREGA** |

Sangolquí, a 03 de marzo de 2021