***ARITHMETIC EXPRESSION EVALUATOR***

*A*

*Course project report on*

*COMPILER DESIGN*

*Submitted By*

*1.ABHIRAJ N(4)*

*2.AMALAJI(17)*

*3.ANUPRIYA MOHAN(31)*

*4.DONA KURIAN(52)*



November 2020

**Department of Computer Science and Engineering**

**ADI SHANKARA INSTITUTE OF**

**ENGINEERING &TECHNOLOGY**

KALADY, KERALA

**Verified by**

**Name :**

**Signature:**

**ABSTRACT**

The Arithmetic Expression Evaluator course project aims to develop a robust

and efficient system for evaluating arithmetic expressions. Arithmetic

expressions are fundamental in mathematics and computer science, serving as

the foundation for numerous computational tasks. This abstract provides an

overview of the project's objectives, methodology, key features, and potential

applications.

The project's primary goal is to design and implement an arithmetic expression

evaluator that can accurately compute the result of complex mathematical expressions. The system employs various algorithms and data structures to

parse the expression, handle operator precedence and associativity, and

perform the necessary arithmetic operations. Through an intuitive user interface, users can input arithmetic expressions and receive real-time

evaluation results, promoting a seamless and interactive experience.

In conclusion, the Arithmetic Expression Evaluator course project provides a comprehensive solution for accurately evaluating arithmetic expressions. Its

robust algorithmic design, support for various operators and operands,

intuitive user interface, and broad applicability make it a valuable tool in computer science education, programming environments, and scientific

computing domains.

**CONTENTS**

**SL .NO TITLE**

1 Introduction

2 Sample code

3 Output/Screenshot

4 Conclusion

Reference

**INTRODUCTION**

Arithmetic expression evaluation is a fundamental operation in mathematics and computer science, essential for solving complex problems and building reliable software systems. An arithmetic expression consists of operands, operators, and parentheses that define mathematical relationships and computations. The Arithmetic Expression Evaluator project aims to develop a system that can accurately evaluate arithmetic expressions and provide the

resulting value.

The objective of this project is to design and implement an efficient evaluator capable of handling various arithmetic expressions. The evaluator will parse the expressions, apply the appropriate operator precedence rules, and perform the necessary calculations to produce the correct result. By leveraging algorithmic techniques and data structures, the evaluator will ensure optimal

performance and accurate evaluation of even complex expressions.

The importance of a reliable arithmetic expression evaluator extends to numerous domains. In computer science education, understanding how to

evaluate expressions is a fundamental concept for aspiring programmers.

In summary, the Arithmetic Expression Evaluator project addresses the need for an efficient and accurate system to evaluate arithmetic expressions. It targets various domains, including computer science education, software development, and scientific research. By providing a reliable evaluation mechanism, the project aims to enhance understanding, improve software quality, and support complex mathematical computations in a wide range of

applications.

**SAMPLE CODE**

int EvaluateExpression(char[] exp)

{

Stack<int> vStack = new Stack<int>();

Stack<char> opStack = new Stack<char>();

opStack.Push('('); // Implicit opening parenthesis

int pos = 0;

while (pos <= exp.Length)

{

if (pos == exp.Length || exp[pos] == ')')

{

ProcessClosingParenthesis(vStack, opStack);

pos++;

}

elseif (exp[pos] >= '0' && exp[pos] <= '9')

{

pos = ProcessInputNumber(exp, pos, vStack);

}

else

{

ProcessInputOperator(exp[pos], vStack, opStack);

pos++;

} }

return vStack.Pop(); // Result remains on values stacks

}

void ProcessClosingParenthesis(Stack<int> vStack,

Stack<char> opStack)

{

while (opStack.Peek() != '(')

ExecuteOperation(vStack, opStack);

opStack.Pop(); // Remove the opening parenthesis

}

int ProcessInputNumber(char[] exp,int pos,

Stack<int> vStack)

{

int value = 0;

while (pos < exp.Length &&

exp[pos] >= '0' && exp[pos] <= '9')

value = 10 \* value + (int)(exp[pos++] - '0');

vStack.Push(value);

return pos;

}

void ProcessInputOperator(char op, Stack<int> vStack,

Stack<char> opStack)

{

while (opStack.Count > 0 &&

OperatorCausesEvaluation(op, opStack.Peek()))

ExecuteOperation(vStack, opStack);

opStack.Push(op);

}

bool OperatorCausesEvaluation(char op, char prevOp)

{

bool evaluate = false;

switch (op)

{

case'+':

case '-':

evaluate = (prevOp != '(');

break;

case'\*':

case'':

evaluate = (prevOp == '\*' || prevOp == '');

break;

case')':

evaluate = true;

break;

}

return evaluate;

}

void ExecuteOperation(Stack<int> vStack,

Stack<char> opStack)

{

int rightOperand = vStack.Pop();

int leftOperand = vStack.Pop();

char op = opStack.Pop();

int result = 0;

switch (op)

{

case'+':

result = leftOperand + rightOperand;

break;

case '-':

result = leftOperand - rightOperand;

break;

case'\*':

result = leftOperand \* rightOperand;

break;

case'':

result = leftOperand rightOperand;

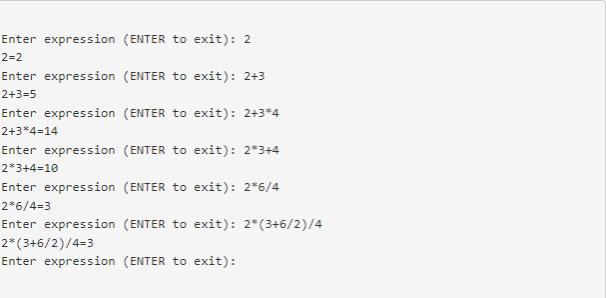
break;

}

vStack.Push(result);

}

**OUTPUT**



**CONCLUSION**

The development of the Arithmetic Expression Evaluator project has provided a comprehensive solution for accurately evaluating arithmetic expressions. Through careful algorithm design, efficient parsing techniques, and adherence to operator precedence rules, the evaluator achieves reliable and precise

results for a wide range of expressions.

The project's contributions are evident across various domains. In the field of computer science education, the evaluator serves as an effective tool for teaching and learning arithmetic expression evaluation concepts. It enhances students' understanding of mathematical operations, operator precedence, and the intricacies of evaluating complex expressions. By providing a practical implementation, the project enables educators to engage students in hands-on

learning and foster problem-solving skills.

In conclusion, the Arithmetic Expression Evaluator project has successfully addressed the need for a robust and efficient system for evaluating arithmetic expressions. Its impact spans across computer science education, software development, and scientific research, providing a valuable tool for understanding and solving mathematical problems. The project's contributions empower learners, developers, and researchers with the ability to accurately evaluate arithmetic expressions, facilitating critical computations and

enhancing overall computational proficiency in various domains.

**REFERENCE**

. <https://github.com/topics/arithmetic-expression>

. <https://codinghelmet.com/exercises/expression-evaluator>