Compiler Design Project

Ahmed Mohamed Mounir, ID: 224643
John Medhat, ID: 224343
Amr Khaled Abdeltawab, ID: 224363

July 18, 2024

Contents

1 Introduction

This document provides a detailed overview of our Compiler Design project. It includes descriptions and code snippets from each phase of the project, along with screenshots illustrating the functionality of our compiler.

2 Phase 1: Lexical Analysis

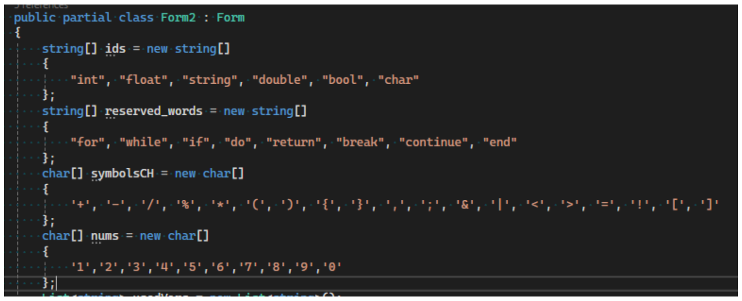
2.1 Introduction

In the first phase, we implemented lexical analysis, which involves reading the source code and breaking it down into tokens. This phase identifies keywords, symbols, numbers, and user-defined variables.

2.2 Code Explanation

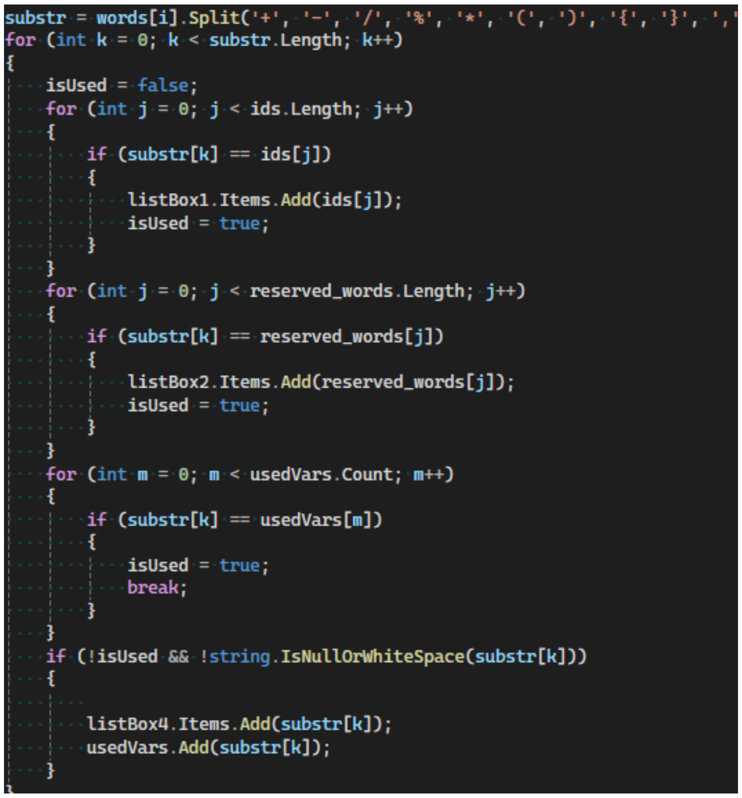
The core functionality is implemented in `Form2.cs`. Below are some important code snippets with explanations.

Token Definitions



```
public partial class Form2 : Form
{
    string[] ids = new string[]
    {
        "int", "float", "string", "double", "bool", "char"
    };
    string[] reserved_words = new string[]
    {
        "for", "while", "if", "do", "return", "break", "continue", "end"
    };
    char[] symbolsCH = new char[]
    {
        '+', '-', '/', '%', '*', '(', ')', '{', '}', ',', ';', '&', '|', '<', '>', '=', '!', '[', ']'
    };
    char[] nums = new char[]
    {
        '1','2','3','4','5','6','7','8','9','0'
    };
```

Text Changed Event



```
substr = words[i].Split('+', '-', '/', '%', '*', '(', ')', '{', '}', ','
for (int k = 0; k < substr.Length; k++)
{
    isUsed = false;
    for (int j = 0; j < ids.Length; j++)
    {
        if (substr[k] == ids[j])
        {
            listBox1.Items.Add(ids[j]);
            isUsed = true;
        }
    }
    for (int j = 0; j < reserved_words.Length; j++)
    {
        if (substr[k] == reserved_words[j])
        {
            listBox2.Items.Add(reserved_words[j]);
            isUsed = true;
        }
    }
    for (int m = 0; m < usedVars.Count; m++)
    {
        if (substr[k] == usedVars[m])
        {
            isUsed = true;
            break;
        }
    }
    if (!isUsed && !string.IsNullOrWhiteSpace(substr[k]))
    {

        listBox4.Items.Add(substr[k]);
        usedVars.Add(substr[k]);
    }
}
```

2.3 Screenshots

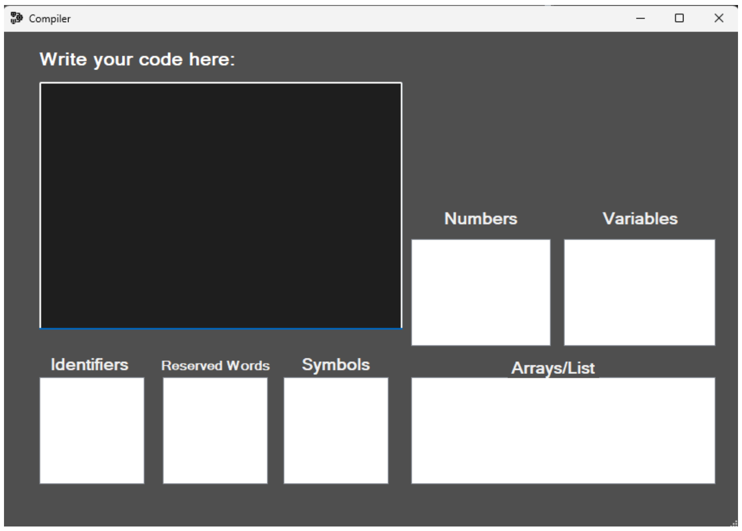


Figure 1: Phase 1 - Lexical Analysis Example

3 Phase 2: Syntax Analysis

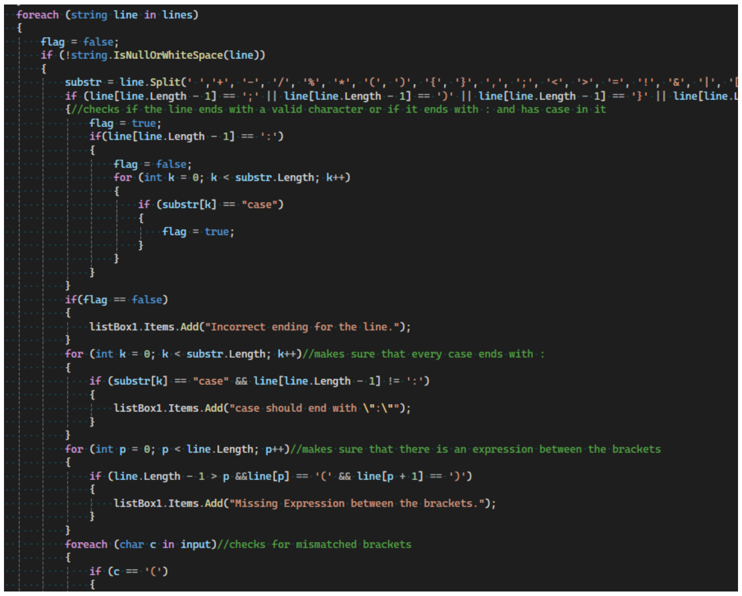
3.1 Introduction

The second phase involves syntax analysis, where the structure of the code is checked according to the grammatical rules of the programming language. This phase ensures that the code adheres to the correct syntax.

3.2 Code Explanation

The core functionality is implemented in `Form3.cs`. Below are some important code snippets with explanations.

Reserved Words Initialization



```
foreach (string line in lines)
{
    flag = false;
    if (!string.IsNullOrWhiteSpace(line))
    {
        substr = line.Split(' ','+', '-', '/', '%', '*', '(', ')', '{', '}', ',', ';', '<', '>', '=', '!', '&', '|', '[
        if (line[line.Length - 1] == ';' || line[line.Length - 1] == ')' || line[line.Length - 1] == '}' || line[line.L
        {//checks if the line ends with a valid character or if it ends with : and has case in it
            flag = true;
            if(line[line.Length - 1] == ':')
            {
                flag = false;
                for (int k = 0; k < substr.Length; k++)
                {
                    if (substr[k] == "case")
                    {
                        flag = true;
                    }
                }
            }
        }
        if(flag == false)
        {
            listBox1.Items.Add("Incorrect ending for the line.");
        }
        for (int k = 0; k < substr.Length; k++)//makes sure that every case ends with :
        {
            if (substr[k] == "case" && line[line.Length - 1] != ':')
            {
                listBox1.Items.Add("case should end with \":\"");
            }
        }
        for (int p = 0; p < line.Length; p++)//makes sure that there is an expression between the brackets
        {
            if (line.Length - 1 > p &&line[p] == '(' && line[p + 1] == ')')
            {
                listBox1.Items.Add("Missing Expression between the brackets.");
            }
        }
        foreach (char c in input)//checks for mismatched brackets
        {
            if (c == '(')
            {
```

3.3 Screenshots

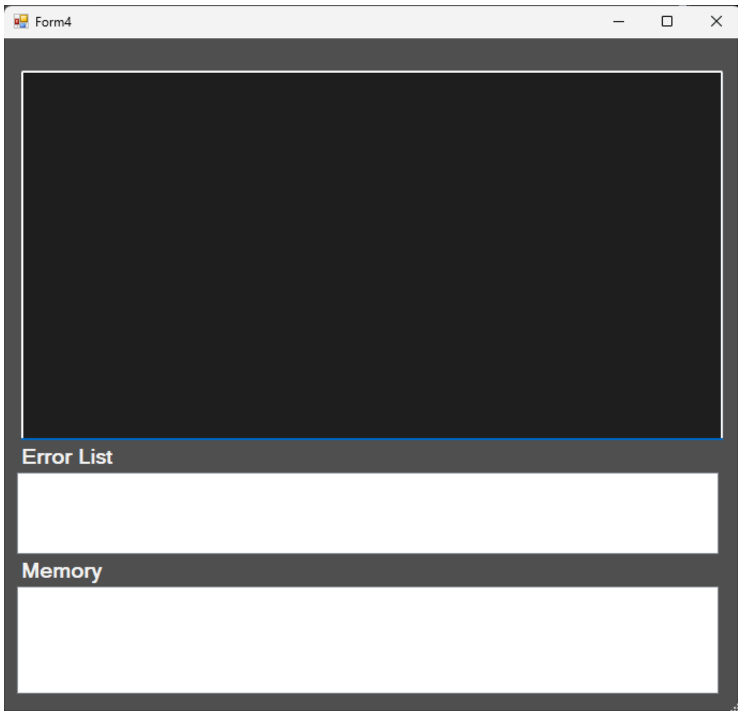


Figure 2: Phase 3 - GUI

4 Phase 3: Semantic Analysis

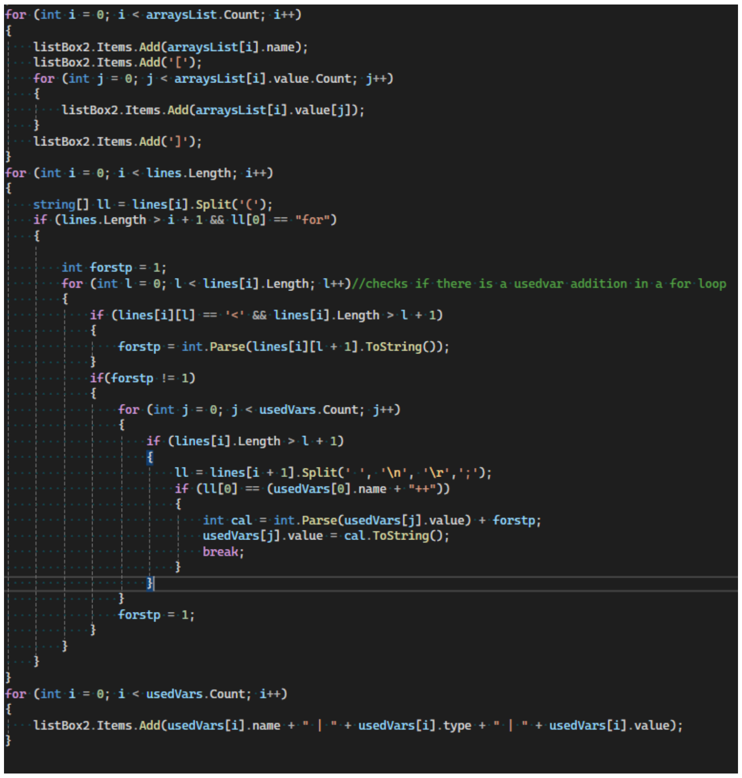
4.1 Introduction

The third phase of our compiler project focuses on semantic analysis. Semantic analysis ensures that the meaning of the code is correct and checks for any logical errors. This phase involves analyzing the context and relationships between different parts of the code.

4.2 Code Explanation

In this phase, we implemented semantic analysis functionalities in `Form4.cs`. Below are some key code snippets with explanations.

Arithmetic Operations



```csharp
for (int i = 0; i < arraysList.Count; i++)
{
    listBox2.Items.Add(arraysList[i].name);
    listBox2.Items.Add('[');
    for (int j = 0; j < arraysList[i].value.Count; j++)
    {
        listBox2.Items.Add(arraysList[i].value[j]);
    }
    listBox2.Items.Add(']');
}
for (int i = 0; i < lines.Length; i++)
{
    string[] ll = lines[i].Split('(');
    if (lines.Length > i + 1 && ll[0] == "for")
    {

        int forstp = 1;
        for (int l = 0; l < lines[i].Length; l++)//checks if there is a usedvar addition in a for loop
        {
            if (lines[i][l] == '<' && lines[i].Length > l + 1)
            {
                forstp = int.Parse(lines[i][l + 1].ToString());
            }
            if(forstp != 1)
            {
                for (int j = 0; j < usedVars.Count; j++)
                {
                    if (lines[i].Length > l + 1)
                    {
                        ll = lines[i + 1].Split(' ', '\n', '\r',';');
                        if (ll[0] == (usedVars[0].name + "++"))
                        {
                            int cal = int.Parse(usedVars[j].value) + forstp;
                            usedVars[j].value = cal.ToString();
                            break;
                        }
                    }
                }
                forstp = 1;
            }
        }
    }
}
for (int i = 0; i < usedVars.Count; i++)
{
    listBox2.Items.Add(usedVars[i].name + " | " + usedVars[i].type + " | " + usedVars[i].value);
}
```

4.3 Screenshots

Include relevant screenshots to illustrate the semantic analysis phase.

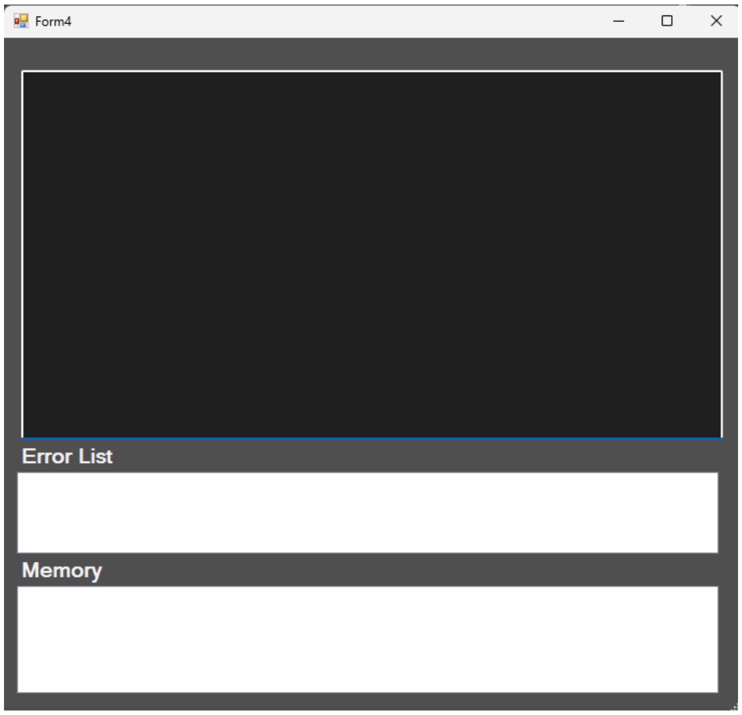


Figure 3: Phase 3 - Semantic Analysis Example

5 Conclusion

This project demonstrates the implementation of a simple compiler with lexical and syntax analysis phases. Each phase is crucial for ensuring that the source code is correctly parsed and validated.