**­**

|  |  |
| --- | --- |
| **Total Marks:** | **04** |
| **Obtained Marks:** |  |

**Finite Automata Theory and Formal Languages**

**Assignment # 04**

**Last date of Submission: 26 Dec 2024**

**Submitted To: Muhammad Nadeem Khokhar**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Student Name: Ubaid Bin Waris**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Reg Number: 2212416 \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

***Instructions****: Copied or shown assignments will be marked zero. Late submissions are not entertained in any case.*

**Question**

Design and implement a Pushdown Automaton (PDA) for the language L = {anbn | n ≥ 0}. Begin by creating a visual representation of the PDA using software tools such as MS Visio or draw.io. The diagram should clearly depict the states, transitions, and stack operations necessary for the PDA to accept strings in the specified language.

Next, write a program in C or C++ to implement the PDA. The program should simulate the PDA's behavior and validate whether input strings belong to the language L. As part of this task, execute the program with the input strings `'aabb'` and `'abbb'`, and determine if they are accepted or rejected by the PDA.

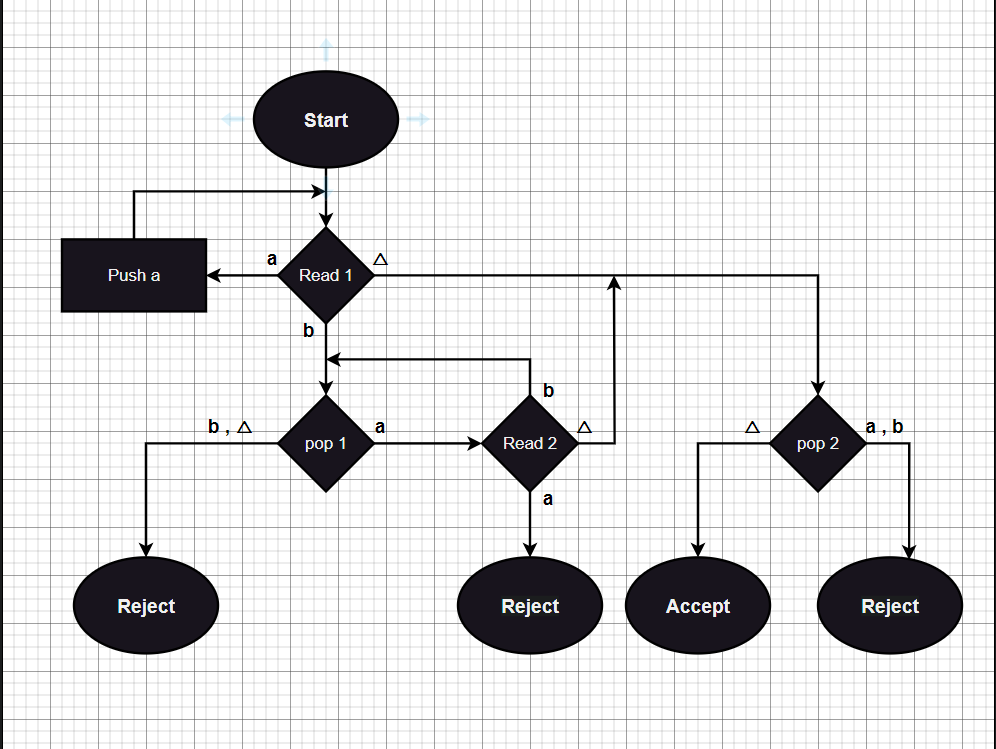
Finally, capture and include a screenshot of the program's runtime output to demonstrate its execution and correctness.

**Note:**

1. Change the filename to your ID, e.g. 2073105.doc
2. Upload the .doc on Google Classroom.
3. Do not use system calls.
4. Make sure that the output screen is complete and does not have colored background.
5. Poor indentation and wrong format will result in deduction of marks.

**Solution**

**Diagram**



**Code**

#include <iostream>

#include <stack>

#include <string>

using namespace std;

string input;

int i = -1;

stack<char> pdaStack;

stack<char> tempStack;

bool isAccepted();

void Read1();

void Read2();

void pop1();

void pop2();

int main()

{

    int choice;

    cout << "Enter a string use '^' fro lembda : ";

    cin >> input;

    cout << "1. fast track\n2. step by step\n";

    cout << "Enter your choice: ";

    cin >> choice;

    cout << "\n";

    switch (choice)

    {

    case 2:

        pdaStack.push('^');  // pushing null value in stack

        input = input + '^'; // adding null value at the end of string

        cout << "Start\n";

        Read1();

        break;

    case 1:

        if (isAccepted())

            cout << "\t\t< < < String Accepted > > >\n";

        else

            cout << "\t\t! ! ! string Rejected ! ! !\n";

        break;

    default:

        cout << "Invalid choice\n";

        break;

    }

}

bool isAccepted()

{

    if (input.length() == 1 && input[0] == '^')

        return true;

    for (int j = 0; j < input.length(); j++)

    {

        if (input[j] != 'a' && input[j] != 'b')

            return false;

        if (input[j] == 'a')

            pdaStack.push('a');

        else if (input[j] == 'b')

            break;

    }

    for (int j = 0; j < input.length(); j++)

    {

        if ((input[j] != 'a' && input[j] != 'b') || pdaStack.empty())

            return false;

        if (input[j] == 'b')

            pdaStack.pop();

    }

    if (pdaStack.empty())

        return true;

    else

        return false;

}

void Read1()

{

    i++;

    if (input[i] == '^' && input.length() == 1)

    {

        cout << "Read1 Stack: " << pdaStack.top() << "\n";

        cout << "String Accepted because its length is one and containing null value\n";

        cout << "\n\t\t< < < String Accepted > > >\n";

        return;

    }

    tempStack = pdaStack;

    cout << "Read1 Stack: ";

    while (!tempStack.empty())

    {

        cout << tempStack.top() << " ";

        tempStack.pop();

    }

    cout << "\n";

    if (input[i] == 'a')

    {

        pdaStack.push('a');

        Read1();

    }

    else if (input[i] == 'b')

        pop1();

    else if (input[i] == '^')

        pop2();

    else

    {

        cout << "Strng contain invalid characters\n";

        cout << "\n\t\t! ! ! string Rejected ! ! !\n";

    }

}

void Read2()

{

    tempStack = pdaStack;

    cout << "Read2 Stack: ";

    while (!tempStack.empty())

    {

        cout << tempStack.top() << " ";

        tempStack.pop();

    }

    cout << "\n";

    i++;

    if (input[i] == 'a')

    {

        cout << "String is Rejected because string contains a after b\n";

        cout << "Language cannot accept a after b\n";

        cout << "\n\t\t! ! ! string Rejected ! ! !\n";

        return;

    }

    else if (input[i] == 'b')

        pop1();

    else if (input[i] == '^')

        pop2();

    else

    {

        cout << "Strng contain invalid characters\n";

        cout << "\n\t\t! ! ! string Rejected ! ! !\n";

    }

}

void pop1()

{

    char ch = pdaStack.top();

    pdaStack.pop();

    tempStack = pdaStack;

    cout << "pop1  Stack: ";

    while (!tempStack.empty())

    {

        cout << tempStack.top() << " ";

        tempStack.pop();

    }

    cout << "\n";

    if (ch == 'b' || ch == '^')

    {

        cout << "String is Rejected because string have more number of b's then a's\n";

        cout << "Language cannot accept more number of b's then a's\n";

        cout << "\n\t\t! ! ! string Rejected ! ! !\n";

        return;

    }

    else if (ch == 'a')

        Read2();

    else

    {

        cout << "Strng contain invalid characters\n";

        cout << "\n\t\t! ! ! string Rejected ! ! !\n";

    }

}

void pop2()

{

    char ch = pdaStack.top();

    pdaStack.pop();

    tempStack = pdaStack;

    cout << "pop2  Stack: ";

    while (!tempStack.empty())

    {

        cout << tempStack.top() << " ";

        tempStack.pop();

    }

    cout << "\n";

    if (ch == 'a' || ch == 'b')

    {

        cout << "String is Rejected because string contains a or b after null value or more a's then b's\n";

        cout << "Language cannot accept a and b after null value or more a's then b's\n";

        cout << "\n\t\t! ! ! string Rejected ! ! !\n";

        return;

    }

    else if (ch == '^')

    {

        cout << "String is accepted because it contains equal numbers of a's and b's\n";

        cout << "\n\t\t< < < String Accepted > > >\n";

    }

    else

    {

        cout << "Strng contain invalid characters\n";

        cout << "\n\t\t! ! ! string Rejected ! ! !\n";

    }

}

**Output**

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated