CSE313: Design of Language Processor

Practical File

Practical - 1

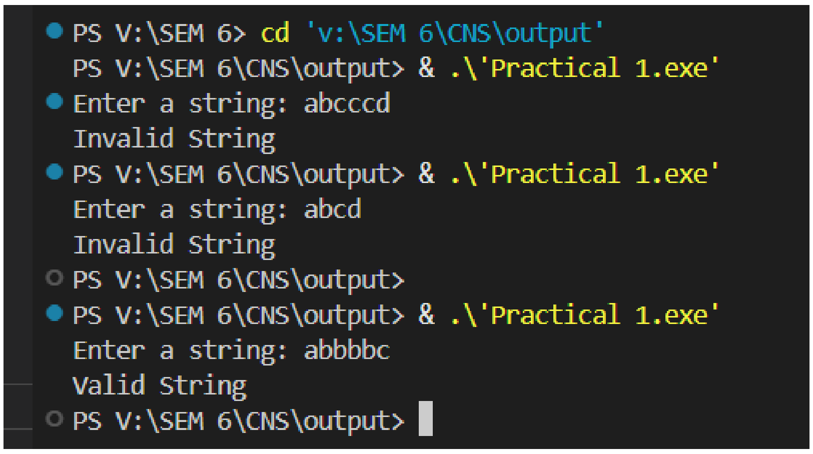
Aim:

To simulate a rule-driven input validation engine similar to compiler front-end scanning, develop a C program that validates user-entered strings according to the formal language: L = $\{a\ b^n\ c \mid n \geq 0\}$. The system should accept a runtime input, analyze the structural correctness of the string without using built-in regex libraries, and determine whether the input conforms to the defined regular language. The solution must show the ability to translate formal language specification → algorithmic logic → executable validation.

Code:

```
#include <stdio.h>
int main() {
    char str[100];
    int i = 0;
    printf("Enter a string: ");
    scanf("%s", str);
    if (str[i] != 'a') {
        printf("Invalid String\n");
        return 0;
    }
    i++;
    while (str[i] == 'b') {
        i++;
    }
    if (str[i] == 'c' && str[i + 1] == '\0') {
        printf("Valid String\n");
    } else {
        printf("Invalid String\n");
    }
    return 0;
}
```

Output (Screenshots):



```
PS V:\SEM 6> cd 'v:\SEM 6\CNS\output'
PS V:\SEM 6\CNS\output> & .\'Practical 1.exe'
Enter a string: abcccd
Invalid String
PS V:\SEM 6\CNS\output> & .\'Practical 1.exe'
Enter a string: abcd
Invalid String
PS V:\SEM 6\CNS\output>
PS V:\SEM 6\CNS\output> & .\'Practical 1.exe'
Enter a string: abbbbc
Valid String
PS V:\SEM 6\CNS\output>
```

Key Questions & Answers:

1. What does the pattern ab*c represent?
   A string that starts with a, ends with c, and may have any number of b in between (including zero).

2. Why can't we compare strings directly to solve this?
   Because the number of b is not fixed.

3. Which automata can recognize this language?
   Finite Automaton (DFA/NFA)

4. What happens if extra characters appear after c?
   The string becomes invalid.

5. What does b* mean?
   * (Kleene Star) means 0 or more occurrences

Applications:

1. Token validation
2. Software Verification

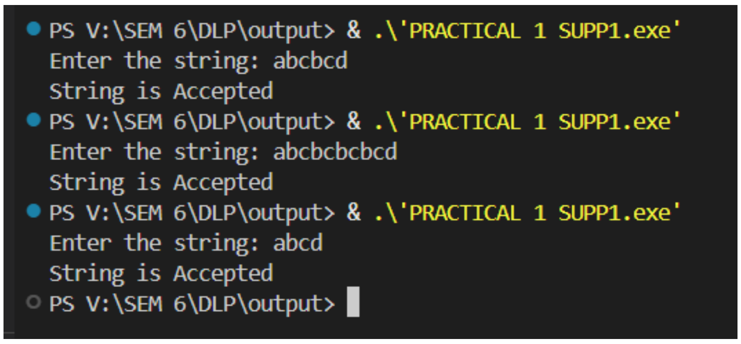
Supplementary Problems (If Applicable):

1. **Modify the program to validate strings that follow the pattern a(bc)*d.**

   Code:

```
#include <stdio.h>
#include <string.h>
int main()
{
  char s[100];
  int i = 0;

  printf("Enter the string: ");
  scanf("%s", s);

  if (s[i] != 'a')
  {
    printf("String is Rejected\n");
    return 0;
  }
  i++;
  while (s[i] == 'b' && s[i + 1] == 'c')
  {
    i += 2;
  }
  if (s[i] == 'd' && s[i + 1] == '\0')
  {
    printf("String is Accepted \n");
  }
  else
  {
    printf("String is Rejected\n");
  }
  return 0;
}
```

Output:



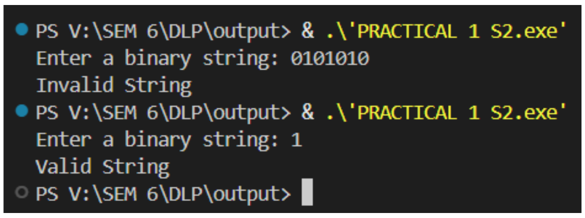
```
PS V:\SEM 6\DLP\output> & .\'PRACTICAL 1 SUPP1.exe'
Enter the string: abcbcd
String is Accepted
PS V:\SEM 6\DLP\output> & .\'PRACTICAL 1 SUPP1.exe'
Enter the string: abcbcbcbcd
String is Accepted
PS V:\SEM 6\DLP\output> & .\'PRACTICAL 1 SUPP1.exe'
Enter the string: abcd
String is Accepted
PS V:\SEM 6\DLP\output>
```

2. **Design a program that checks the pattern (01) *1 (odd number of 1's).**

Code:

```
#include <stdio.h>

int main() {
   char str[100];
   int i = 0;

   printf("Enter a binary string: ");
   scanf("%s", str);

   while (str[i] == '0' && str[i + 1] == '1') {
      i += 2;
   }

   if (str[i] == '1' && str[i + 1] == '\0') {
      printf("Valid String\n");
   } else {
      printf("Invalid String\n");
   }

   return 0;
}
```

Output:



```
PS V:\SEM 6\DLP\output> & .\'PRACTICAL 1 S2.exe'
Enter a binary string: 0101010
Invalid String
PS V:\SEM 6\DLP\output> & .\'PRACTICAL 1 S2.exe'
Enter a binary string: 1
Valid String
PS V:\SEM 6\DLP\output>
```