CSA1361 -
Theory of Computation

Name: k. Sree Durga Geethika
Reg no: 192111772.
sub code: CSA1361.

Observation Notes

(1) Aim: write a C-Program for DFA for the given language representing strings that start with a and end with 'a'

Program:

```
#include <stdio.h>
#include <string.h>
main()
{
 char s[10]; A = 'a';
 int c=0, n;
 Print f("Enter the string :");
 Scanf("%s", s);
 n = strlen(s);
 for (int i=0; i<n; i++)
  {
   if [(s[i]=='a' || s[i]=='b')
  {
   c=0;
  }
  else
   {
    c=1;
   Print ("INVALID");
   break;
   }
 }
if (·-·')
 {
  if (s[n] = = 'a' && s[n-1] = = 'a')
   {
    Print f ("The string was accepted");
   }
  else
   {
    Print f ("NOT DFA");
   }
 }
}
```

output:

```
Enter the string: aaaabbaa
The string was accepted.
```

② Aim: write a C-program for DFA for the given language representing strings that starts with 0 and end with 1.

Program:

```
# include <stdio.h>
# include <string.h>
# define MAX 100
int main
{
  char str[max]; f='a';
  int i;
 Print f ("Enter the string to be checked");
scanf ("%s", str);
int length = strlen(str);
if ( str[0] == '0' && str [length-1] == '1')
 {
   printf ("Accepted");
  }
 else
  {
   printf ("Rejected");
   return 0;
    }
 }
```

output:

```
   Enter the string to be checked : 00011
   Accepted.
```

③ Aim: write a C-program to check whether a given string belongs to the language defined by a CFG

$S \rightarrow 0A1 \quad A \rightarrow 0A|1A|\epsilon$

Program:

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

```
printf("Enter a string to check");
scanf("%s", s);
  l = strlen(s);
  flag=1;
  for (i=0; i<l; i++)
   {
    if (s[i]!='0' && s[i]!='1')
    {
     flag=0;
    }
   }
  if (flag!=1)
  printf("String is not valid\n");
  if (flag==1)
  ~~printf("~~
  {
  if ((s[0]=='0' && s[l-1]=='1')
     printf("string is accepted");
    else
    printf("string is not accepted\n");
  }
 }
```

<u>output</u>:

```
Enter a string to check: 001
String is accepted.
```

(1) Aim: write a c-program to check whether a given string belongs to the language defined by a CFG S → 0S0|1S1|0|1|ε

Program:

```
# include< stdio.h>
# include < string.h>
# define max 100
int main ( )
 {
 char str[max] = 'a';
 int i;
 printf ("Enter the string to be checked:");
 scanf ("%S", str);
  int length = strlen(str);
  if (str[0] == str[length-1] && str[i] == str[
                                           length-2])
  {
   printf ("Accepted");
   }
  Else
   {Print f("rejected );
   return 0;
     }
   }
```

output:

```
Enter the string to be checked.00000
Accepted.
```

(5) **Aim:** write a C program to check whether a given string belongs to the language defined by CFG S → 0S0|A A → 1A1|ε

Program:

```
#include <stdio.h>
# include <string.h>
int main( )
{
Char s[100];
int i, flag, flag1, a, b;
int l, Count1, Count2;
print f(" Enter a string to check:");
scanf("%s", s);
 l= strlen(s);
 flag=1;
 for (i=0; i<l; i++)
  {
  if (s[i]!='0'&& s[i]!='1')
   {
    flag=0;
    }
   }
   if (flag!=1)
   Print f("String is not valid\n");
   if(flag==1)
   {
    i=0; Count1=0;
   while (s[i]=='0')
```

```
{
count1++;
  i++;
  }
  while(s[i]=='1')
   {
    i++;
    }
   flag1=1;
   Count2=0;
   while(i<l)
   {if(s[i]=='0')
   {
   Count2++;
   }
   else
    {
    flag1=0;
    }
    i++;
   }
   if(flag1==1)
    {
     if(Count1==Count2)
      {
   Printf("the string satisfies the condition
                on m0n\n");
  Printf("string accepted");
   }
   else
    {
     Printf("The string doesnot satisfy the
            condition on 1m0n\n");
Printf("string not accepted");
    }
   }
  }
 }
```

output:

```
        Enter a string to check : 00001111
  The string doesnot satisfy the Condition 0^n1^n:
string not accepted.
```

(6) Aim: write a c-program to check whether a given string belongs to the language defined by a CFG $S \to 0S|\epsilon$

Program:

```
#include <stdio.h>
#include <string.h>
int main()
{
char S[100];
int i, flag, flag1, a, b;
int l, Count1, Count2;
Printf(" Enter a string to check:");
Scanf("%s", s);
l= strlen(s);
flag =1;
for (i=0; i<l; i++)
 {
 if (s[i]!='0' && s[i]!='1')
 {
```

```
flag=0;
}
}
if(flag==1)
 {
 i=0; Count1=0;
 while(s[i]=='0')
 {
 Count1++;
  i++;
  }
   while(s[i]=='1']
   {
     i++;
    }
    flag1=1;
     Count2=0;
      while(i<0)
      {
       if(s[i]=='0')
       {
       Count2++;
 }
 else
 {
 flag1=0;
 }
 i++;
 }
 if (flag1==1)
 {
 if(Count1==Count2)
 {
 Printf("The string satisfies the Condition");
printf("string accepted");
 }
 else
 {
 printf("The string doesnot satisfies the
Condition")
 printf("not accepted");
   }
  }
 }
```

output :

```
    Enter a string to check :000
  The string doesnot satisfy the Condition.
```

(7.) Aim: write a c-program to check whether a given string belongs to the language defined by a CFG $S \rightarrow A1A \ , A \rightarrow 0A|A|\varepsilon$

Program:

```
#include <stdio.h>
#include <string.h>
int main()
 {
 char s[100];
 int i, flag, flag1;
 int l;
 printf("Enter a string to check");
 scanf("%s", s);
 l= strlen(s);
 l= strlen(s);
```

```
flag=1;
for (i=0;i<l ;i++)
{if (s[i]!='0' && s[i]!='1')
 {
  flag=0;
  }
 }
 if (flag==1)
printf("string is valid\n");
else
printf("string is not valid\n");
if (flag==1)
 {
 flag1=0;
 for(i=0; i<l-2 ;i++)
 {if (s[i]=='1')
  {
   if (s[i+1]=='0' && s[i+2] =='1')
   {
   flag1=1;
   printf("substring 101 doesnot exist. string
   not accepted\n");
  }
 }
```

output: Enter a string to check: 11111
string is valid
substring 101 doesnot exist

(5) **Aim:** write a C-program to simulate a NFA the given language representing strings that start with b and end with a

Program:

```
# include < stdio.h>
# include <string.h>
int main ()
{
 char s[10], B='b', A='a';
  int n, c=0;
  printf ("Enter the string.");
   scanf ("%s", s);
    n = strlen(s);
    for (int i=0; i<n; i++)
     {
      if (s[i]=='b'|| s[i]=='a')
       {
         c=0;
         }
        else
         {
         c=1;
          printf ("invalid");
          break;
         }
       }
     if (c==0)
      {
       if (s[0]==B && s[n-1]== A)
       {
        printf (" The string was accepted");
        }
else
{
 printf("not DFA");
 }
 }
}
```

output: Enter the string:100
  invalid.

(9.) Aim: write a C program to simulate a NFA for the given language representing strings that start with 0 & end with 1.

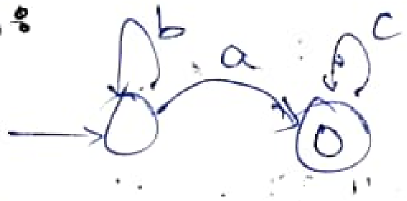
program:

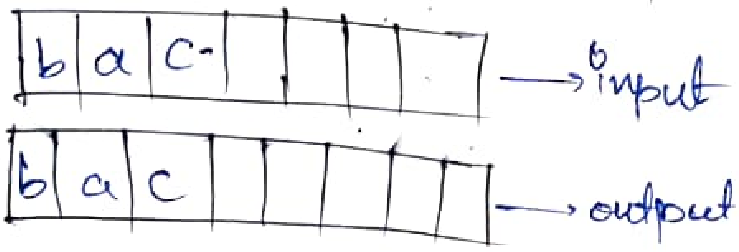
```
#include<stdio.h>
#include<string.h>
int main()
{
char s[10], B='0', A='1';
int n, c=0;
printf("Enter the string:");
scanf("%s", s);
n= strlen(s);
for(int i=0; i<n; i++)
  {  if(s[i]=='0'|| s[i]=='1'
    {
     c=0;
    }
    else
     {
      c=1;
printf("invalid");
break;
}
}
if (c==0)
{
 if (s[0]==B && s[n-1]==A)
 {
  printf("the string was accepted");
 }
 else
 {
  printf("Not NFA");
 }
}
}
```

output: Enter the string = 11111
not NFA.

(12.) Aim: To design a DFA for input string 'a', 'b' and 'bac' using simulator.

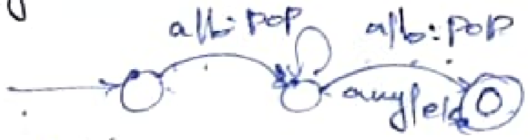
Diagram:





13. Aim: Design a PDA for input string aabb using simulator.

Diagram:



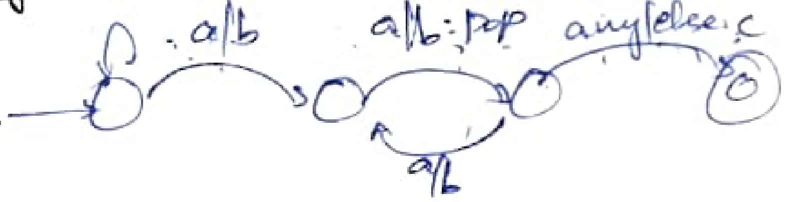
Input

a	a	b	b			

output

(14) Aim: Design a PDA for $a^n b^{2n}$ using simulator

Diagram:



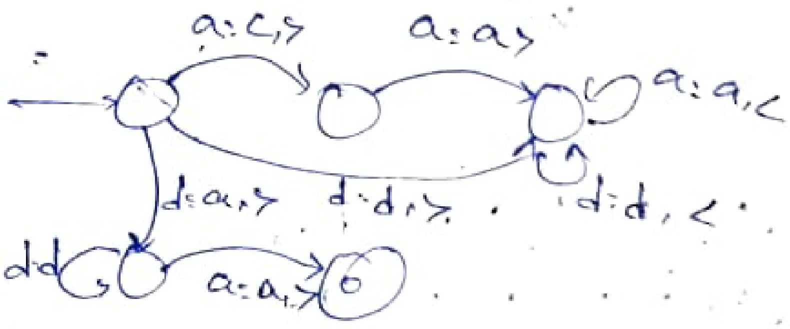
Input

a	a	b	b	b	b

output

				●		

(15) Aim: Design TM for input string $a^n b^n$ using simulator.



Input

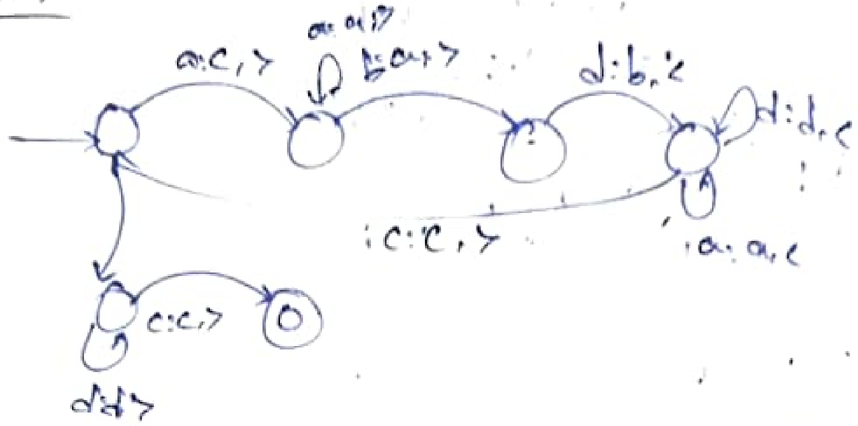
a	a	b	b

output

c	c	d	d	

(16) Aim: Design Tm for input string $a^n b^{2n}$ using simulator.

Diagram:



input

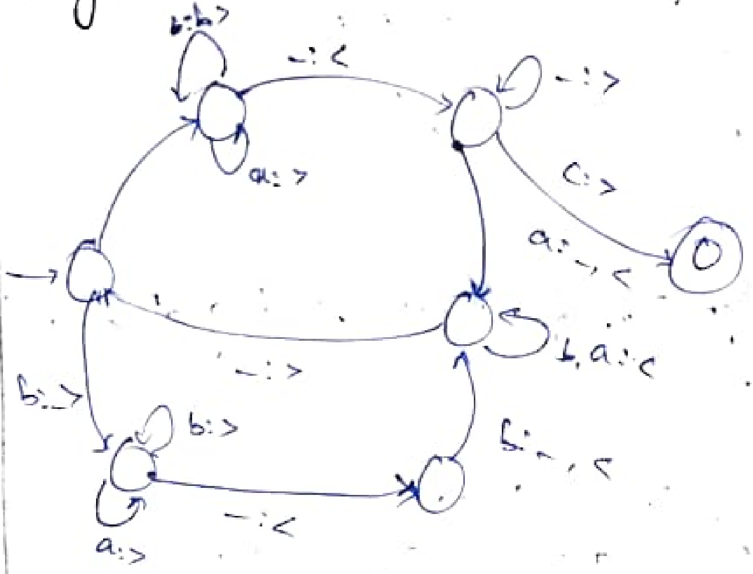
a	a	b	b	b	b

output

c	c	d	d	d	d	✓

17) Aim: Design a Tm for input string palindrome ababa using simulator.

Diagram:



Input

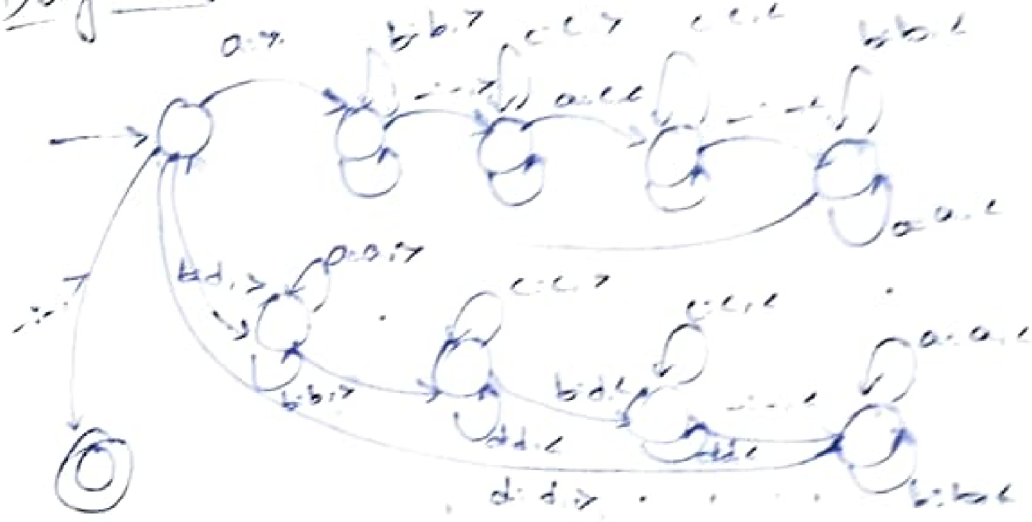
a	b	a	b	a		c

output

a	b	a	b	a		c

Aim: Design Tm to accept the output string ww using simulator.

Diagram:



input

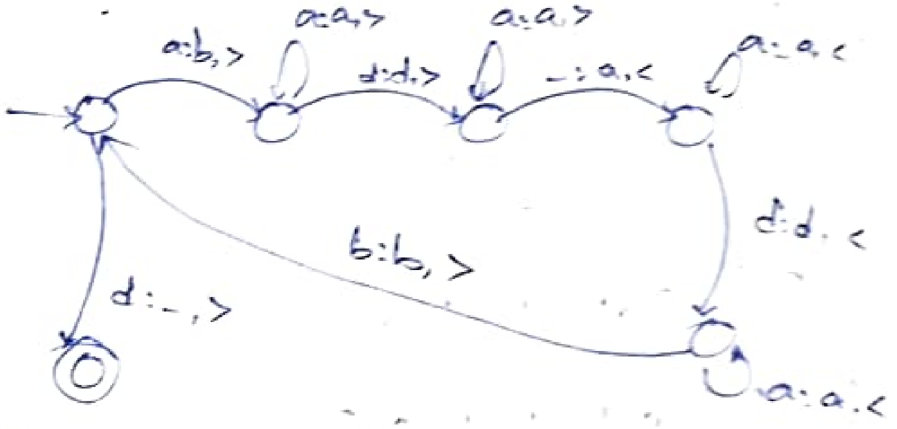
c	d	c		c	d	c

output

c	d	c		c	d	c

(a) Aim: Design a Tm for addition of 'aa' and 'aaa' using simulator.

Diagram:



input

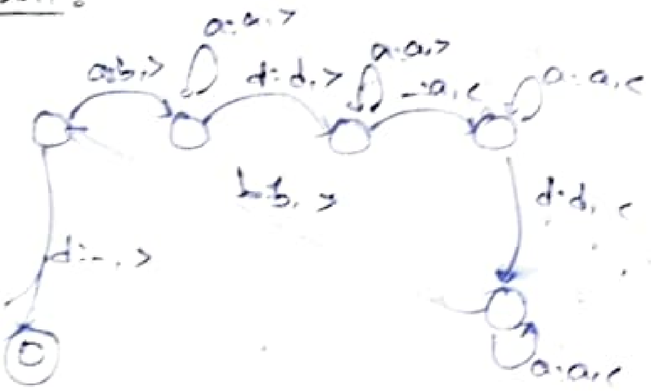
		a	a		d	a	a	a

output

		b	b		a	a	a	a	a

(53) Aim: Design a Tm for subtraction of a a a ... using simulator.

Diagram:



input

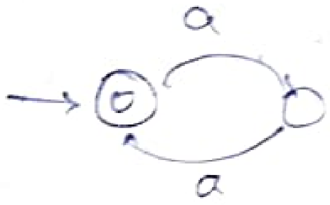
		a	a	d	a	a	a

output

			bb		a	a	a	a	a

(54) Aim: Design a Tm for even number of a's using simulator.

Diagram:



input

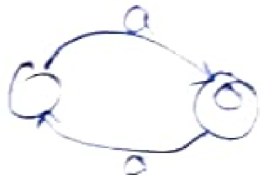
a	a	a	a

output

a	a	a	a	✓	

(55) Aim: Design DFA for odd number of a's using simulator.

Diagram:



Input

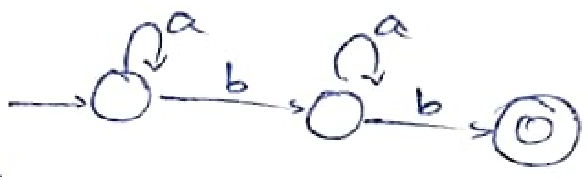
a	a	a

output

a	a	a	✓

(23) **Aim:** Design DFA to accept the string end with b over set {a,b} w= aaabab using simulator.

Diagram:



input

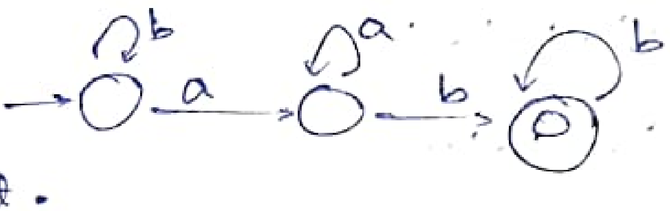
a	a	a	b	a	b

output

a	a	a	b	a	b	✓

(24) **Aim:** Design a DFA to accept the string having 'ab' as substring over the set {a,b}

Diagram:



input:

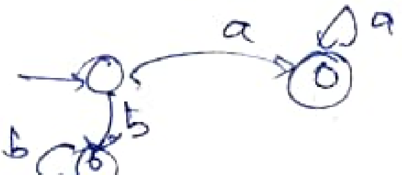
a	a	a	b

output:

a	a	a	b	✓

(25) **Aim:** Design DFA using simulator to accept the string start with a or b the set {a,b}

Diagram:



input

a	a	

output

a	a	v	

(28) Aim: Design Tm for accept the input string with ab over set {a,b} w= abbaabab

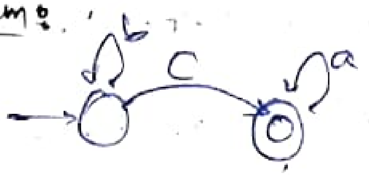
Diagram:

Input

output.

(29) Aim: Design DFA with using simulator accept the input string 'bc', "c' ε "bcaa'

Diagram:



Input

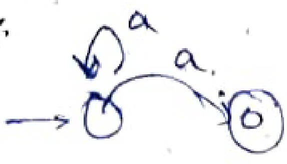
b	c	a	a	a

output

b	c	a	a	a	v

(30) Aim: Design NFA to accept any number being simulator.

Diagram:



input

a	a	a	a	a