Ahsanullah University of Science & Technology

Department of Computer Science & Engineering

Course No.	**:**	CSE 4130
Course Name	**:**	Formal Languages and Compilers Lab

Assignment No.	**:**	05

Submitted By:

Name	:	Mohammad Shah Alam
ID No.	:	17-01-04-012
Session	:	Spring - 2020
Section	:	A (A1)

QUESTION :

Implement the following CFG in the way shown above.

A → aXd
X → bbX
X → bcX
X → ▯

ANSWER:

```
#include<stdio.h>
#include<string.h>
char str[30];
int len,i=0,f=0;
void X()
{
    if(len-1 == i)
    {
        i++;
        f = 1;
        return;
    }
    else
    {
        if(str[i] == 'b')
        {
            i++;
            if(str[i] == 'b' || str[i] == 'c')
            {
                i++;
                X();

            }
        }
        else
        {
            f = 0;
            return;
        }
    }
}
void A()
{
    if(str[i] == 'a')
    {
        i++;
        X();
        if(f==1)
        {
            if(str[i-1] == 'd')
                f = 1;
            else
                 f = 0;

            return;
        }
    }
}
int main(void)
{
   printf("CFG: \n");
    printf("A -> aXd\nX -> bbX | bcX | epsilon\n");     ///Valid : ad,abbd,abbbbd,abcd,abcbbd
,etc.

   char c[100];
   FILE *fptr,*p2;
   p2 = fopen("output_file.txt","w");
   if ((fptr = fopen("input.txt","r")) == NULL){
       printf("Error! opening file");

       // Program exits if the file pointer returns NULL.
```

```
      exit(1);
  }
  while(fscanf(fptr,"%s",&str)>0)
  {
       i = 0;
       len = strlen(str);
       if(len>=1)
           {
               A();
           }
       else
           fprintf(p2,"Go find another string %s please!\n",str);
       if(len == i && f == 1)
           fprintf(p2,"Your will %s is valid..\n",str);
        else
           fprintf(p2,"Again you have come to disturb!! your will %s is not valid!\n",str);
  }
  fclose(fptr);
  fclose(p2);
  return 0;
}
```

QUESTION :

2. A CFG to describe the syntax of simple arithmetic expressions may look like the one that follows:

<Exp>→<Term> + <Term> | <Term> - <Term> | <Term>

<Term>→<Factor> * <Factor> | <Factor> / <Factor> | <Factor>

<Factor>→(<Exp>) | ID | NUM

ID → a|b|c|d|e

NUM→ 0|1|2|...|9

ANSWER:

```
#include<bits/stdc++.h>
using namespace std;
```

```
string s;
int i = 0;
bool F();

bool ID()
{
    if(s[i]=='a' || s[i]=='b' || s[i]=='c' || s[i]=='d' || s[i]=='e')return true;
    else return false;
}

bool NUM()
{
    if(s[i]>='0' && s[i]<='9')return true;
    else return false;
}

bool T()
{
    if(F())
    {
        i++;
        if(s[i]=='*' || s[i]=='/' )
        {
            i++;
            if(F())return true;
            else
            {
                i--;
                return false;
            }
        }
        else
        {
            i--;
            return true;
        }
    }
    else return false;
}

bool E()
{
    if(T())
    {
        i++;
        if(s[i]=='+' || s[i]=='-')
        {
            i++;
            if(T())return true;
            else
            {
                i--;
                return false;
            }
        }
        else
        {
            i--;
            return true;
        }
    }
    else return false;
}

bool F()
{
    if(s[i]=='(')
    {
        i++;
        if(E())
        {
            i++;
            if(s[i]==')')return true;
            else
            {
                i--;
                return false;
            }
        }
        else
        {
            i--;
            return false;
        }
    }
    else if(ID())return true;
    else if(NUM())return true;
    else return false;
}
```

```
int main()
{
    cout<<"<Exp> --> <Term> + <Term> | <Term> - <Term> | <Term>\n";
    cout<<"<Term> --> <Factor> * <Factor> | <Factor> / <Factor> |
<Factor>\n";
    cout<<"<Factor> --> ( <Exp> ) | ID | NUM\n";
    cout<<"ID --> a|b|c|d|e\n";
    cout<<"NUM --> 0|1|2|3|4|5|6|7|8|9\n";
 char ch;


        FILE *file1,*file2;
    file1=fopen("input2.cpp","r");
    file2=fopen("output2.cpp","w");
    if(!file1)
        cout<<"File can't be opened"<<endl;
    else
    {
        while((ch=fgetc(file1))!=EOF)
        {

            if(ch!='\n')
            {
                s+=ch;

            }
        else
        {
                if(E()){


                    cout<<"valid\n";
                    fputc('v',file2);
                    fputc('a',file2);
                    fputc('l',file2);
                    fputc('i',file2);
                    fputc('d',file2);
                    fputc('\n',file2);
                     i = 0;
                     s.clear();
                }
                else{


                    cout<<"invalid\n";
                      fputc('i',file2);
                    fputc('n',file2);
                    fputc('v',file2);
                    fputc('a',file2);
                    fputc('l',file2);
                    fputc('i',file2);
                    fputc('d',file2);
                    fputc('\n',file2);
                     s.clear();
                    i = 0;
                }
```

```
            }
        }

    i = 0;
        if(E()){


                    cout<<"valid\n";
                    fputc('v',file2);
                    fputc('a',file2);
                    fputc('l',file2);
                    fputc('i',file2);
                    fputc('d',file2);
                    fputc('\n',file2);

                     s.clear();
                }
                else{


                    cout<<"invalid\n";
                      fputc('i',file2);
                    fputc('n',file2);
                    fputc('v',file2);
                    fputc('a',file2);
                    fputc('l',file2);
                    fputc('i',file2);
                    fputc('d',file2);
                    fputc('\n',file2);
```

```
                        s.clear();

                }

    }


}
//(a+2)*(a-2)
```

QUESTION :

3.Implement the following grammar in C.

```
<stat>→<asgn_stat>□<dscn_stat>□<loop_stat>
<asgn_stat>→id = <expn>
<expn>→<smpl_expn> <extn>
<extn>→<relop> <smpl_expn> | □
<dcsn_stat>→ if (<expn> ) <stat> <extn1>
<extn1>→ else <stat> | □
<loop_stat>→while (<expn>) <stat>□for (<asgn_stat> ; <expn> ; <asgn_stat> ) <stat>
<relop>→ ==□!=□<=□>=□>□<
```

Note: <smpl_expn> can be implemented using the materials demonstrated in this session.

ANSWER:

```
#include<stdio.h>
#include<string.h>
#include<iostream>
using namespace std;

char str[100];
int f = 0,i = 0,l;
```

```
void stat();
void asgn_stat();
void dscn_stat();
void loop_stat();
void expn();
void smpl_expn();
void extn();
void relop();
void extn1();
void E();
void T();
void F();


void loop_stat()
{
    if(str[i] == 'w' || str[i+1] == 'h'|| str[i+2] == 'i'|| str[i+3] == 'l'|| str[i+4] ==
'e')
    {
        i = i+5;
        if(str[i] == '(')
        {
            i++;
            expn();
            if(str[i] == ')')
            {
                i++;
                stat();
                if(i==l)
```

```
                    return;
                else
                {
                    f = 0;
                    return;
                }

            }
            else
                return;
        }
        else
        {
            f = 0;
            return;
        }
    }
    else if(str[i] = 'f' || str[i+1] == 'o' || str[i+2] == 'r')
    {
        i = i+3;
        if(str[i] == '(')
        {
            i++;
            asgn_stat();
            if(str[i] == ';')
            {
                i++;
                expn();
                if(str[i] == ';')
                {
                    i++;
                    asgn_stat();
                    if(str[i] == ')')
                    {
                        i++;
                        stat();
                        if(i==l)
                            return;
                        else
                        {
                            f = 0;
                            return;
                        }
                    }
                    else
                    {
                        f = 0;
                        return;
                    }

                }
                else
                {
                    f = 0;
                    return;
                }
            }
            else
            {
                f = 0;
                return;
            }
        }
        else
        {
            f = 0;
            return;
        }
    }
    else
    {
        f = 0;
        return;
    }
}


void extn1()
{
    if((l-1) == i)
    {
        f = 1;
        i++;
        return;
    }
    else
    {
        if(str[i] == 'e' && str[i+1] == 'l' && str[i+2] =='s' || str[i+3] == 'e')
        {
            i=i+4;
            f=0;
            stat();
            if(f == 1)
                return;
            else
            {
                f = 0;
                return;
            }
        }
        else
        {
            f = 0;
            return;
        }
    }
}


void dscn_stat()
{
    if(str[i] == 'i')
    {
        i++;
        if(str[i] == 'f')
        {
            i++;
            if(str[i] == '(')
            {
                i++;
                expn();
                if(str[i] == ')')
                {
                    i++;
                    stat();
                    if(i==l)
                        return;
                    else
                    {
                        if(f == 1)
                        {
                            extn1();
                        }
                        else
                            return;
                    }

                }
            }
        }
    }
    else
    {
        f = 0;
        return;
    }
}
void F()
{
    if(isdigit(str[i]))
    {
        i++;
        f = 1;
        return;
    }
    else if(str[i] == 'a' || str[i] == 'b' || str[i] == 'c' || str[i] == 'd')
    {
        i++;
        f = 1;
        return;
    }
    else if(str[i] == '(')
    {
        i++;
        E();
        i++;
        if(str[i] == ')')
        {
            f = 1;
            return;
        }
    }
}


void T()
{
    F();
    if(i==l)
        return;
    if(i<l-1)
    {
        if(str[i] == '*' || str[i] == '/')
        {
            i++;
            F();
        }
        else if(f == 1)
        {
            return;
        }
    }

}

void E()
{
    T();

    if(i == l)
        return;
    if(i < l-1)
    {
        if(str[i] == '+' || str[i] == '-')
        {
            i++;
            T();
        }
        else if(f == 1)
        {
            return;
        }
    }

}

void smpl_expn()
{
    E();
    if(f == 1 && l==i)
    {
        return;
    }
    else
        return;
}

void relop()
{
    if(str[i] == '=')
    {
        i++;
        if(str[i] == '=')
        {
            f  = 1;
            return;
        }
        else
        {
            f = 0;
            return;
        }
    }
    else if(str[i] == '!')
    {
        i++;
        if(str[i] == '=')
        {
           f  = 1;
           return;
        }
        else
        {
            f = 0;
            return;
        }
    }
    else if(str[i] == '<')
    {
        i++;
        f = 1;
        if(str[i] == '=')
        {
           f  = 1;
           return;
        }
        else
        {
            return;
        }
    }
    else if(str[i] == '>')
    {
        i++;
        f = 1;
        if(str[i] == '=')
        {
            i++;
            f  = 1;
            return;
        }
        else
        {
            return;
        }
    }
    else if(str[i] == '>')
    {
        i++;
        f  = 1;
        return;
    }
    else if(str[i] == '<')
    {
        i++;
        f  = 1;
        return;
    }
    else
    {
        f = 0;
        return;
    }

}


void extn()
{
    if((l-1) == i)
    {
        f = 1;
        i++;
        return;
    }
    else
    {
        relop();
        if(f == 1)
        {
            smpl_expn();
            if(l == i)
                return;
        }
        else
            return;
    }
}
```

```
void expn()
{
    smpl_expn();
    if(l == i)
    {
        return;
    }
    else
    {

        if(f == 1)
        {
            extn();
            return;
        }

    }


}



void asgn_stat()
{
    if(str[i] == 'a' || str[i] == 'b' || str[i] == 'c'|| str[i] == 'd'|| str[i] == 'e')
    {
        i++;
        if(str[i] == '=')
        {
            i++;
            expn();
            if(f ==1 && i==l)
            {
                return;
            }
            else
            {
                f=1;
                return;
            }

        }

    }
    else
    {
        f = 0;
        return;
    }
}



void stat()
{
    int as = 0;
    asgn_stat();
    as = 1;
    if(f ==1 && (l==i))
    {
        return;
    }
    else if(f==1)
        return;

    if(as == 1 && f == 0)
    {
        //i=0;
        dscn_stat();
        if(f == 0)
        {
            //i = 0;
            loop_stat();
        }

    }
}

int main()
{
    cout <<"CFG: " <<endl;
    cout << "<stat> -> <asgn_stat> | <dscn_stat> | <loop_stat>" <<endl
<< "<asgn_stat> -> id = <expn>"<<endl<<
            "<expn> -> <smpl_expn> <extn>"<<endl<<"<extn> -> <relop>
<smpl_expn> | epsilon"<<endl<<"<dcsn_stat> -> if (<expn> ) <stat>
<extn1>"<<endl<<
            "<extn1> -> else <stat> | epsilon "<<endl<<"<loop_stat> -> while
(<expn>) <stat> | for (<asgn_stat> ; <expn> ; <asgn_stat> ) <stat>"
<<endl<<
            "<relop> -> == | != | <= | >= | >| <"<<endl;
    //VALID:
    //b=1,c=a,b=a*3,

//if(a)b=1,if(a>5)b=1elsec=2,if(a>5)b=5elseif(a<5)c=5,if(a>5)b=5elseif(a<5)
c=5elsed=5,

//while(a)b=1,while(a>=b)a=1,while(a>=b)if(a>5)b=5elseif(a<5)c=5elsed=5

//for(a=0;a<2;c=1)b=3,for(a=0;a<5;a=a+1)b=a+b,for(a=0;a<5;a=a+1)if(a>
5)b=a,for(a=0;a<5;a=a+1)if(a>5)b=aelsec=b,for(a=0;a<5;a=a+1)if(a>5)b=a
elseif(a<5)c=b;

    cout <<endl;
    cout <<"Enter string: "<<endl;
    cin >> str;
    cout <<endl;

    l = strlen(str);
    if(l >= 1 )
        stat();
    else
        cout <<"invalid string"<<endl;
    if(l == i && f == 1 )
        cout <<"Valid string"<<endl;
    else
    {
        cout <<"invalid string"<<endl;
    }
    return 0;
}
```