PARSER PROJECT

HTML Tags for Unordered and Ordered list (with Nesting)

SUBMITTED BY –

SALONI DOBHAL (34)

YASHIKA GOLA (47)

SANDEEP KUMAR (52)

SUBMITTED TO –

DR. ANKIT RAJPAL

PROGRAM

File Name - project.l

```
%{
        #include "y.tab.h"
%}

%x olAttributeList ulAttributeList attributeListItem

attributeValue          (\"([^\"]*)\")
whitespace              ([ \t]*)
olTag                   "ol"
ulTag                   "ul"
liTag                   "li"
olAttList               "type"|"start"|"reversed"|"style"
ulAttList               "type"|"style"
AttListItem             "value"

%%
"<"{whiteSpace}{olTag}      {
                                BEGIN(olAttributeList);
                                return START_OL_TAG;
                            }
"<"{whiteSpace}{ulTag}      {
                                BEGIN(ulAttributeList);
                                return START_UL_TAG;
                            }
"<"{whiteSpace}{liTag}      {
                                BEGIN(attributeListItem);
                                return START_LI_TAG;
                            }
```

```
(">")                           {
                                        return yytext[0];
                                }
([a-zA-Z0-9 ])*                 {
                                        return TEXT;
                                }
("</ol>")                       {
                                        return OL_END_TAG;
                                }
("</ul>")                       {
                                        return UL_END_TAG;
                                }

("</li>")                       {
                                        return LI_END_TAG;
                                }
<olAttributeList>{
        {olAttList}{whiteSpace}[=]      {
                                                return ATTRIBUTE;
                                        }
        {attributeValue}                {
                                                return ATTRIBUTE_VAL;
                                        }
        (">")                           {
                                                BEGIN(INITIAL);
                                                return yytext[0];
                                        }
}
<ulAttributeList>{
        {ulAttList}{whiteSpace}[=]      {
                                                return ATTRIBUTE;
```

```
                                        }
        {attributeValue}                {
                                                return ATTRIBUTE_VAL;
                                        }
        (">")                           {
                                                BEGIN(INITIAL);
                                                return yytext[0];
                                        }
}

<attributeListItem>{
        {AttListItem}{whiteSpace}[=]    {
                                                return ATTRIBUTE;
                                        }
        {attributeValue}                {
                                                return ATTRIBUTE_VAL;
                                        }
        (">")                           {
                                                BEGIN(INITIAL);
                                                return yytext[0];
                                        }
}

.                               {return yytext[0];}
"\n"                            {return NEWLINE;}
%%

int yywrap()
{
        return 1;
}
```

File Name - project.y

```
%{
                #include<stdio.h>
                #include<stdlib.h>
                int yylex();
                int yyerror();
%}

%start START
%token START_OL_TAG START_UL_TAG ATTRIBUTE ATTRIBUTE_VAL TEXT OL_END_TAG
UL_END_TAG START_LI_TAG LI_END_TAG NEWLINE



%%
/*Grammar*/
START           :ORDERED {printf("Syntax is valid\n"); return 0;}
                |UNORDERED {printf("Syntax is valid\n"); return 0;}
                ;
ORDERED         :START_OL_TAG ATT '>' LIST_TAG OL_END_TAG
                ;
UNORDERED       :START_UL_TAG ATT '>' LIST_TAG UL_END_TAG
                ;
LIST_TAG        :LIST_TAG START_LI_TAG ATT '>' BODY LI_END_TAG
                |{}
                ;
BODY            :BODY TEXT
                |{}
                |ORDERED
```

```
                        |UNORDERED
                        ;
ATT                     :ATT ATTRIBUTE ATTRIBUTE_VAL
                        |{}
                        ;

%%

int yyerror() {
  printf("Syntax is invalid\n");
  return 1;
}

int main()
{
                printf("\nHTML lists with nesting\n");
                printf("Enter HTML code: ");
                yyparse();
                return 0;
}
```

WHAT PARSER CAN DO

1. It can parse Ordered and Unordered Lists with List-Items in HTML syntax.
2. It can parse one or more attributes in <ol>, i.e., *type, start, reversed and style*.
3. It can parse one or more attributes in <ul>, i.e., *type and style.*
4. It can also parse attribute of <li> as well, i.e., *value*.
5. Multiple <li> tags inside ordered and unordered list are also parsed.
6. It can parse for nesting of <ol> and <ul>, inside the <li> tag of <ol> or

LIMITATIONS OF PARSER

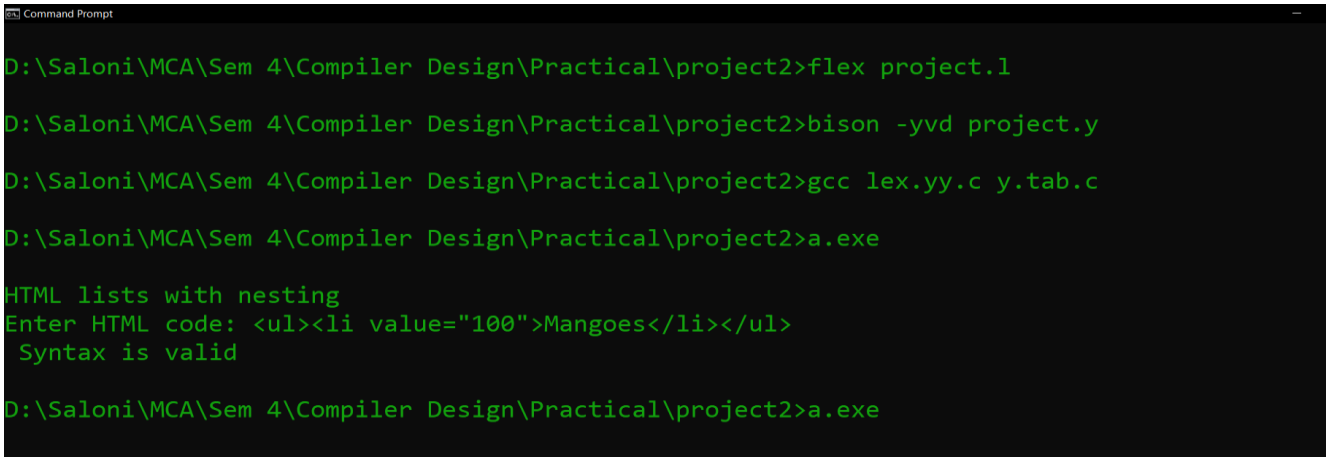
1. Parser gives invalid syntax for space between list-item and ordered list or unordered list.
2. The value attribute inside <li> tag is specified only in the case of ordered list. But parser validates the value attribute in case of unordered list of elements.
   Example - <li value = “100”>Mangoes
   The parser accepts the above syntax, though it is invalid in case of HTML.
3. The parser does not parse the language in capital letters.

OUTPUT

Description –

- The value attribute for <li> tag of unordered list is accepted by parser as valid syntax.

Syntax - <li value = “100”>Mangoes



Description –

- The type attribute of unordered list is specified as “disk”.
- The multiple entries of list-item tag are specified.
- The given syntax is validated by parser.

Syntax - <ul type = “disk”> Mango
banana>

```
D:\Saloni\MCA\Sem 4\Compiler Design\Practical\project2>a.exe

HTML lists with nesting
Enter HTML code: <ul type="disc"><li>Mango</li><li>banana</li></ul>
 Syntax is valid
```

Description –

- The type attribute of ordered list is specified as “1”.
- The nesting of <ul> inside <ol> is specified, including the type attribute.
- The given syntax is validated by parser.

Syntax - <ol type = “1”>
Mangoes
<ul type=”disc”>
RED>
YELLOW>
GREEN> Apples

```
D:\Saloni\MCA\Sem 4\Compiler Design\Practical\project2>a.exe

HTML lists with nesting
Enter HTML code: <ol type="1"><li>Mangoes</li><li><ul type="disc"><li>RED</li><li>YELLOW</li><li>GREEN</li></ul></li><li>Apples</li></o
l>
  Syntax is valid
```

Description –

- The type attribute of ordered list is specified as “1”.
- The style attribute of ordered list is specified as “color:yellow”
- The value attribute of list-item is specified in case of ordered list.
- The nesting of <ul> inside <ol> tag is specified, including the type attribute for <ol>.
- The given syntax is validated by parser.

Syntax - <ol type=”1” style=”color:yellow”><li value=”5”>Mangoes
<ul type=”disc”>
RED
YELLOW
Apples
<ul type =”disc”>
RED
YELLOW
GREEN

```
D:\Saloni\MCA\Sem 4\Compiler Design\Practical\project2>a.exe

HTML lists with nesting
Enter HTML code: <ol type="1" style="color:yellow"><li value="5">Mangoes</li><li><ul type="disc"><li>RED</li><li>YELLOW
li></ul></li><li>Apples</li><li><ul type="disc"><li>RED</li><li>YELLOW</li><li>GREEN</li></ul></li></ol>
     Syntax is valid

D:\Saloni\MCA\Sem 4\Compiler Design\Practical\project2>
```

Description –

- The <ol> closing syntax is not correct.
- Hence, the syntax is invalidated by parser.

Syntax -

```
D:\Saloni\MCA\Sem 4\Compiler Design\Practical\project2>a.exe

HTML lists with nesting
Enter HTML code: <ol><li></li><ul>
Syntax is invalid

D:\Saloni\MCA\Sem 4\Compiler Design\Practical\project2>
```

Description –

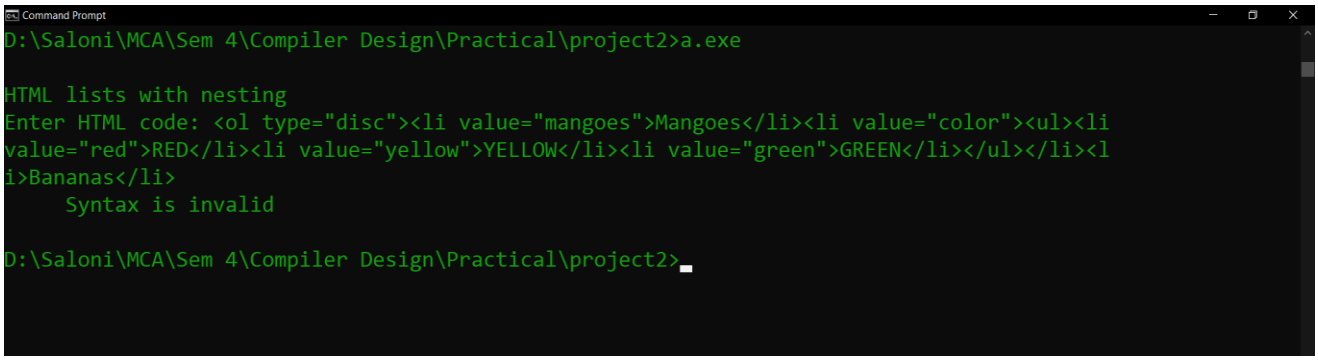
- The attribute of <li> tag is specified as *type,* but it defined as *value* only.
- Hence, the parser will give that syntax is invalid

Syntax - <li type=”100”>Mangoes

Description –

- The nesting of <ul> under <ol> is specified.
- The type attribute of for ordered is specified.
- The value attribute for all <li> tags is specified, except one.
- The closing of <ol> tag is not done.
- Hence, the syntax is invalidated by the parser.

Syntax - <ol type=”disc”><li value=”mangoes”> Mangoes <li value=”color”>
<li value=”red”>RED
<li value=”yellow”>YELLOW
<li value=”green”>GREEN
Banana



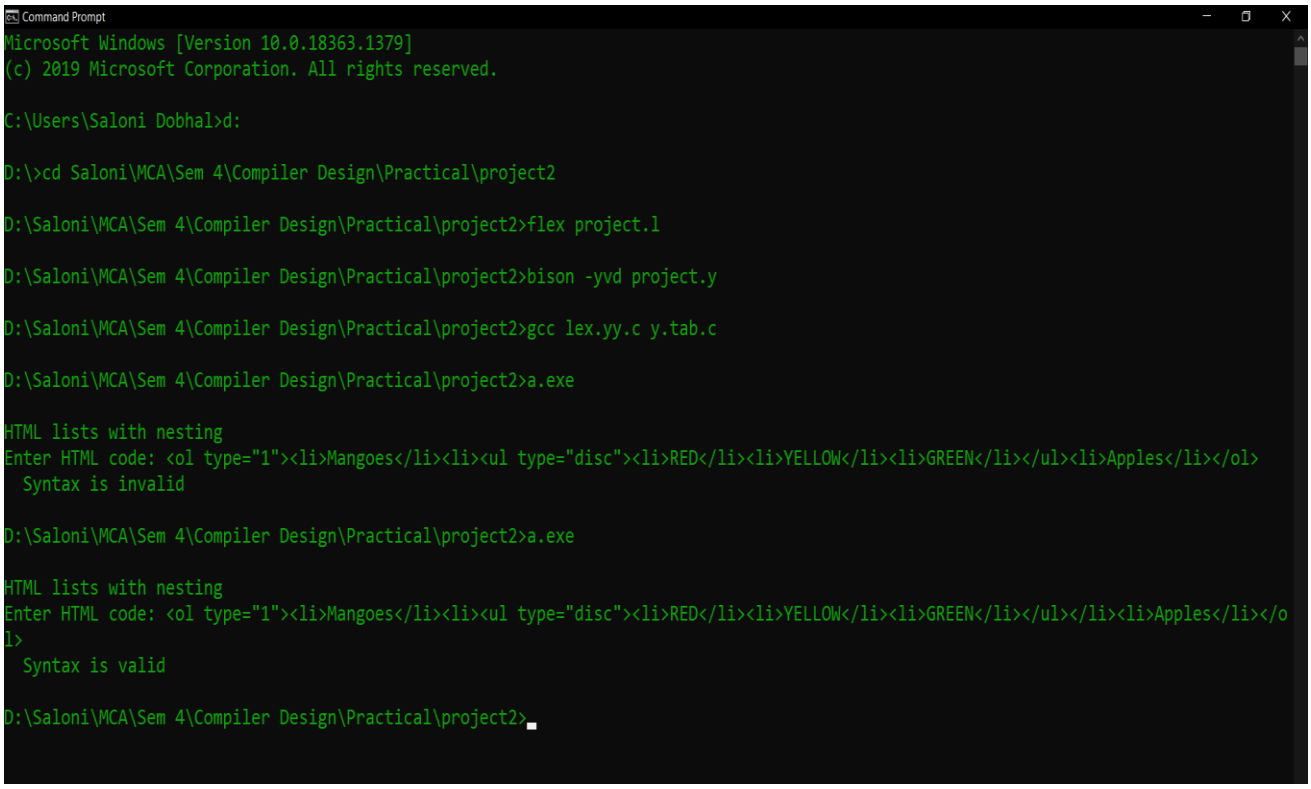
Description –

- The type attribute of both <ol> and <ul> is specified in Code-1 and Code-2.
- The <ul> is nested inside <li> tag of <ol> in Code-1 and Code-2.
- The <li> tag under <ol> where nesting of <ul> is done, is not closed in Code-1.

- Hence, Code-1 is invalidated by the parser.
- The <li> tag under <ol> where nesting of <ul> is done, is closed in Code-2.
- Hence, Code-2 is validated by the parser.

Syntax –

- Code 1 –
  <ol type=”1”>
Mangoes
<ul type=”disc”>
RED
YELLOW
GREEN
Apples
- Code 2-
  <ol type=”1”>
Mangoes
<ul type=”disc”>
RED
YELLOW
GREEN
Apples



REFERENCES –

https://www.w3schools.com/html/html_lists_ordered.asp

https://www.w3schools.com/html/html_lists_unordered.asp

https://github.com/whogeek/biflex-html-parser

https://stackoverflow.com/questions/22527608/begininitial-in-flex-parser

https://stackoverflow.com/questions/2706839/unrecognized-rule-in-lex