/* A10: Suppose an image is encodes as an n×m matrix **M** of "light intensifies." $\mathbf{M}_{ij}$ is a number from 0 to 15, with 0=black and 15=white. **M** may be stored row by row, with rows terminated by newline (n) characters. Call this string $v_M$. For example, if **M** is

| | | | |
|---|---|---|---|
| 0 | 0 | 2 | 6 |
| 0 | 1 | 4 | 7 |
| 1 | 8 | 8 | 6 |

then $v_M$ is *0026 n 0147 n 1886*. We can also encode **M** by its differences along rows, $d_M$. For Example, for **M** above $d_M$ is *0+2+4 n +1+3+3 n+70-2*. If we replace positive numbers by + and negative numbers by - in $d_M$ we get the sequence of changes, $d'_M$. In this case, $d'_M$=0++ n +++ n +0-. Suppose a '*feature*' is defined to be a nonempty sequence of increasing value of intensity, followed by zero to three unchanging value of intensity followed by at least one decreasing value of intensity, all on one row.

a. Write a **LEX** program to find maximal features in $d'_M$ and surround them by parentheses. A maximal feature is not a proper substring of any other feature.
b. Display the longest maximal feature of each row. */

**File: A10_1.l**

```
digit [0-9]
newline [\n]
char [a-zA-Z_]
%{
      /* This program is divided into two lex programs. This
program is for encoding the numbers into +,-,0 notation
         Input file is read and the encoded string is written
into another textfile which acts as input for the second lex
program.
      */

      int i;
      // File pointer for output file.
      FILE* fd;
%}

%%
({digit})* {
      // if it is a digit stream, encode.
      char prev='\0';
      // In every iteration the previous value is remembered and
the difference is taken and appropriate symbol is retrieved.
      for(i=0;i<yyleng;i++)
      {     if(yytext[i]==' ')continue;
            if(yytext[i]!='\n'&&prev!='\0')
            {
                  int ans=yytext[i]-prev;
                  // If difference is 0 the encode it as '0', if
difference is positive encode as '+' and '-' otherwise.
```

```
                if(ans==0){fprintf(fd,"0");}
                else if(ans>0)fprintf(fd,"+");
                else fprintf(fd,"-");

            }
            else if(yytext[i]=='\n')
            {
                fprintf(fd,"\n");
                prev='\0';continue;
            }
            prev=yytext[i];
      }
}
{newline} {fprintf(fd,"\n");}
%%
int main(int argc, char const *argv[])
{
      yyin=fopen("./A10_input.txt","r");
      fd=fopen("./A10_text_in_plus.txt","w");
      yylex();
      return 0;
}
```

---

**File: A10_2.l**

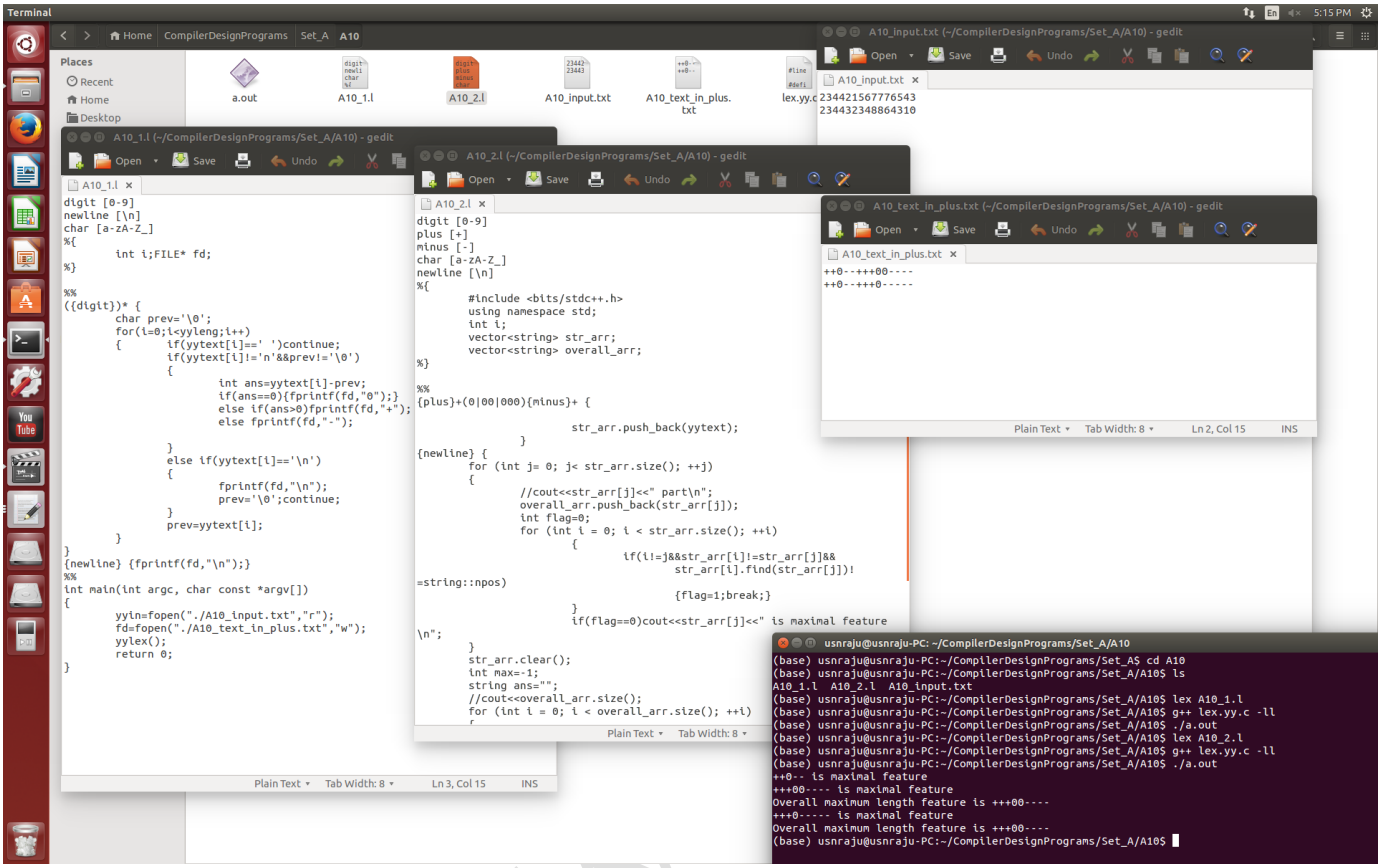
```
digit [0-9]
plus [+]
minus [-]
char [a-zA-Z_]
newline [\n]
%{
      // This is the second lex program. It extracts the required
features.
      #include <bits/stdc++.h>
      using namespace std;
      int i;
      vector<string> str_arr;
      vector<string> overall_arr;
%}

%%
{plus}+(0|00|000){minus}+ {
                // Found a feature in a row so store it in an
array.
                str_arr.push_back(yytext);
            }
{newline} {
      // End of a row so find the maximal feature for that row.
      for (int j= 0; j< str_arr.size(); ++j)
      {
```

```
            //cout<<str_arr[j]<<" part\n";
            overall_arr.push_back(str_arr[j]);
            int flag=0;
            /* Iterate through the entire array and check whether
the current feature is a maximal feature or not,
               i.e check if it is a substring of other feature, if
no then it is a maximal feature for that row.
            */
            for (int i = 0; i < str_arr.size(); ++i)
                  {
                        // Substring check
                        if(i!=j&&str_arr[i]!=str_arr[j]&&

      str_arr[i].find(str_arr[j])!=string::npos)
                              {flag=1;break;}
                  }
                  if(flag==0)cout<<str_arr[j]<<" is maximal
feature\n";
      }
      // Row is done, clear the array.
      str_arr.clear();
      int max=-1;
      string ans="";
      //cout<<overall_arr.size();

      // Among the features find the one with the highest length,
which is the overall maximum.
      for (int i = 0; i < overall_arr.size(); ++i)
      {
            int temp=overall_arr[i].length();;

      if(temp>max){max=overall_arr[i].length();ans=overall_arr[i]
;}
      }
      cout<<"Overall maximum length feature is "<<ans<<endl;
}

0*|{plus}*|{minus}* { }
%%
int main(int argc, char const *argv[])
{
      yyin=fopen("./A10_text_in_plus.txt","r");
      yylex();
      return 0;
}
```

---

**File: A10_input.txt**

```
234421567776543
234432348864310
```


Terminal
5:15 PM
Home
CompilerDesignPrograms
Set_A
A10
Places
Recent
Home
Desktop
a.out
A10_1.l
A10_2.l
A10_input.txt
A10_text_in_plus.txt
lex.yy.c
A10_input.txt (~/CompilerDesignPrograms/Set_A/A10) - gedit
Open
Save
Undo
234421567776543
234432348864310
A10_1.l (~/CompilerDesignPrograms/Set_A/A10) - gedit
digit [0-9]
newline [\n]
char [a-zA-Z_]
%{
int i;FILE* fd;
%}
%%
({digit})* {
char prev='\0';
for(i=0;i<yyleng;i++)
{ if(yytext[i]==' ')continue;
if(yytext[i]!='n'&&prev!='\0')
{
int ans=yytext[i]-prev;
if(ans==0){fprintf(fd,"0");}
else if(ans>0)fprintf(fd,"+");
else fprintf(fd,"-");
}
else if(yytext[i]=='\n')
{
fprintf(fd,"\n");
prev='\0';continue;
}
prev=yytext[i];
}
}
{newline} {fprintf(fd,"\n");}
%%
int main(int argc, char const *argv[])
{
yyin=fopen("./A10_input.txt","r");
fd=fopen("./A10_text_in_plus.txt","w");
yylex();
return 0;
}
Plain Text
Tab Width: 8
Ln 3, Col 15
INS
A10_2.l (~/CompilerDesignPrograms/Set_A/A10) - gedit
digit [0-9]
plus [+]
minus [-]
char [a-zA-Z_]
newline [\n]
%{
#include <bits/stdc++.h>
using namespace std;
int i;
vector<string> str_arr;
vector<string> overall_arr;
%}
%%
{plus}+(0|00|000){minus}+ {
str_arr.push_back(yytext);
}
{newline} {
for (int j= 0; j< str_arr.size(); ++j)
{
//cout<<str_arr[j]<<" part\n";
overall_arr.push_back(str_arr[j]);
int flag=0;
for (int i = 0; i < str_arr.size(); ++i)
{
if(i!=j&&str_arr[i]!=str_arr[j]&&
str_arr[i].find(str_arr[j])!
=string::npos)
{flag=1;break;}
}
if(flag==0)cout<<str_arr[j]<<" is maximal feature
\n";
}
str_arr.clear();
int max=-1;
string ans="";
//cout<<overall_arr.size();
for (int i = 0; i < overall_arr.size(); ++i)
Plain Text
Tab Width: 8
A10_text_in_plus.txt (~/CompilerDesignPrograms/Set_A/A10) - gedit
++0--+++00----
++0--+++0-----
Plain Text
Tab Width: 8
Ln 2, Col 15
INS
usnraju@usnraju-PC: ~/CompilerDesignPrograms/Set_A/A10
(base) usnraju@usnraju-PC:~/CompilerDesignPrograms/Set_A$ cd A10
(base) usnraju@usnraju-PC:~/CompilerDesignPrograms/Set_A/A10$ ls
A10_1.l A10_2.l A10_input.txt
(base) usnraju@usnraju-PC:~/CompilerDesignPrograms/Set_A/A10$ lex A10_1.l
(base) usnraju@usnraju-PC:~/CompilerDesignPrograms/Set_A/A10$ g++ lex.yy.c -ll
(base) usnraju@usnraju-PC:~/CompilerDesignPrograms/Set_A/A10$ ./a.out
(base) usnraju@usnraju-PC:~/CompilerDesignPrograms/Set_A/A10$ lex A10_2.l
(base) usnraju@usnraju-PC:~/CompilerDesignPrograms/Set_A/A10$ g++ lex.yy.c -ll
(base) usnraju@usnraju-PC:~/CompilerDesignPrograms/Set_A/A10$ ./a.out
++0-- is maximal feature
+++00---- is maximal feature
Overall maximum length feature is +++00----
+++0----- is maximal feature
Overall maximum length feature is +++00----
(base) usnraju@usnraju-PC:~/CompilerDesignPrograms/Set_A/A10$