ASSIGNMENT NO1

Title: Implementation of Conflation Algorithm to generate document representative of a text file.

Code:

Conflation.c

```
#include<stdio.h>
#include<conio.h>
char
sw[50][10]={"our","we","what","or","by","it.","and","can","on","so","even","if","that","in","be","to","it","is","this",
"the","a","an","was","were","he","she","you","will","are","have","when","off","has","had","for","of","because","xx
"};
char
stm[50][10]={"able","fulness","ousness","ational","tional","alize","ful","ness","sses","ing","ed","s",".",",","ion","yy
"};
char token[30]=" ";
int temp=0;
FILE *fp=NULL;
int i,flag=0,flag1=0,k,j;
int len;

typedef struct index
{
		int no;
		char string[20];
		int frequency;
}index;

index i1[100];

void lower_case()
{
		temp=token[0];
		if(temp<=90)
		{
				token[0]=temp+32;
		}
}

void removesw()
{
		flag=0;
		lower_case();
		for(i=0;i<len;i++)
		{
				if(strcmp(token,sw[i])==0)
				{
						flag=1;
						break;
				}
```

```
        }
        if(flag==0)
        {
                token[0]=temp;
                strcpy(i1[k].string,token);
                k++;
        }
}

void stem()
{
  int len1,l=0;
  char *ptr=NULL;
  for(j=0;j<k;j++)
  {
        i=0;
        flag1=0;
        len1=0;
        while(i1[j].string[i]!='\0')
        {
                len1++;
                i++;
        }
        if(i1[j].string[len1-1]=='.')
        {
                i1[j].string[len1-1]='\0';
        }
        for(i=0;strcmp(stm[i],"yy")!=0;i++)
        {
                *ptr=strstr(i1[j].string,stm[i]);
                if(*ptr!=NULL)
                {
                  flag1=1;
                  for(l=0;stm[i][l]!='\0';l++)
                  {
                        if(i1[j].string[strlen(i1[j].string)-strlen(stm[i])+l]!=stm[i][l])
                        {
                                flag1=0;
                                break;
                        }
                  }
                  break;
                }
        }
        if(flag1==1)
        {
                if(strlen(i1[j].string)-strlen(stm[i])>0)
                {
                        i1[j].string[strlen(i1[j].string)-strlen(stm[i])]='\0';
                }
        }
  }
}
void duplicate()
{
  int count,l=0,temp1;
```

```
    for(j=0;j<k;j++)
    {
            if(strcmp(i1[j].string,"null")!=0)
            {
                count=1;
                for(i=j+1;i<k;i++)
                {
                        if(strcmp(i1[j].string,i1[i].string)==0)
                        {
                                strcpy(i1[i].string,"null");
                                count++;
                        }
                }
                i1[j].no=l;
                l++;
                i1[j].frequency=count;
            }
    }

    for(i=0;i<k;i++)
    {
            if(strcmp(i1[i].string,"null")==0)
            {
                        for(j=i;j<k;j++)
                        {
                                if(strcmp(i1[j].string,"null")!=0)
                                {
                                        temp=i1[i].no;
                                        i1[i].no=i1[j].no;
                                        i1[j].no=temp;

                                        strcpy(token,i1[i].string);
                                        strcpy(i1[i].string,i1[j].string);
                                        strcpy(i1[j].string,token);

                                        temp1=i1[i].frequency;
                                        i1[i].frequency=i1[j].frequency;
                                        i1[j].frequency=temp1;

                                        break;
                                }
                        }
            }
    }
}

void calls(char fname[20])
{
            char freq=' ';
            len=0,k=0;

            fp=fopen(fname,"r");

            if(fp==NULL)
            {
                        printf("\nCant read input file");
```

```
        getch();
        exit(0);
}
else
{
        while(strcmp(sw[i],"xx")!=0)
        {
                len++;
                i++;
        }
        while(!feof(fp))
        {
           fscanf(fp,"%s",token);
           removesw();
        }
}
fclose(fp);
printf("---------------------------------------------");
printf("\nThe contents of %s file\n\n",fname);
printf("---------------------------------------------\n");
fp=fopen(fname,"r");
while(!feof(fp))
{
        fscanf(fp,"%s",token);
        printf("%s ",token);
}
printf("\n\n");
getch();
fclose(fp);

printf("----------------------------------------------");
printf("\n\nThe contents of file after removing Stop words\n");
printf("----------------------------------------------\n");
for(i=0;i<k;i++)
{
        printf("%s ",i1[i].string);
}
printf("\n\n");
getch();
stem();
printf("----------------------------------------------");
printf("\nThe contents of file after removing Suffix words\n");
printf("----------------------------------------------\n");
for(i=0;i<k;i++)
{
  printf("%s ",i1[i].string);
}
printf("\n\n");
getch();
duplicate();
printf("----------------------------------------------");
printf("\n\nThe contents of the file after removing Duplicates\n");
printf("----------------------------------------------\n");
for(i=0;i<k&&strcmp(i1[i].string,"null")!=0;i++)
{
        printf("%s ",i1[i].string);
        }
        getch();
        printf("\n\n");
        printf("-----------------------------------------------");
        printf("\n\nDOCUMENT REPRESENTATIVE  for %s \n\n",fname);
        printf("-----------------------------------------------");
        printf("\nNo.\tKeyword\t\tFrequency\n\n\n");
        printf("-----------------------------------------------");
        printf("\n");
        fp=fopen("Doc_rep.txt","w");
        for(i=0;i<k&&strcmp(i1[i].string,"null")!=0;i++)
        {
                printf("%-2d\t%-15s\t  %-10d\n\n",i1[i].no,i1[i].string,i1[i].frequency);
                fputc(' ',fp);
                fputs(i1[i].string,fp);
                fputc(' ',fp);
                freq=i1[i].frequency+48;
                fputc(freq,fp);

        }
        fclose(fp);
        printf("-----------------------------------------------");
}

void main()
{
        char fname[20];
        clrscr();

        printf("\nEnter Document name:- ");
        scanf("%s",fname);
        strcat(fname,".txt");

        calls(fname);
        getch();

}
```

Input: input.txt
Output:

```
Enter Document name:- input
-----------------------------------------------
The contents of input.txt file

-----------------------------------------------
Sometimes called Remote Wake-up, Wake on LAN is technology that allows someone t
o turn on a network computer remotely by sending a special data packet. Even if
the computer is turned off, the network adapter is still listening on the networ
k, so when the special packet arrives, the network adapter can turn on the compu
ter progress. progress.

-----------------------------------------------

The contents of file after removing Stop words
```

```
----------------------------------------------
Sometimes called Remote Wake-up, Wake LAN technology allows someone turn network
 computer remotely sending special data packet. computer turned off, network ada
pter still listening network, special packet arrives, network adapter turn compu
ter progress. progress.
----------------------------------------------
The contents of file after removing Suffix words
----------------------------------------------
Sometime call Remote Wake-up Wake LAN technology allow someone turn network comp
uter remotely send special data packet computer turn off network adapter still l
isten network special packet arrives, network adapter turn computer progres prog
res

----------------------------------------------

The contents of the file after removing Duplicates
----------------------------------------------
Sometime call Remote Wake-up Wake LAN technology allow someone turn network comp
uter remotely send special data packet off adapter still listen arrives, progress

----------------------------------------------
0     Sometime        1

1     call            1

2     Remote          1

3     Wake-up         1

4     Wake            1

5     LAN             1

6     technology      1

7     allow           1

8     someone         1

9     turn            3

10     network        4

11     computer       3

12     remotely       1

13     send           1

14     special        2

15     data           1

16     packet         2

17     off            1
```

```
18    adapter       2
19    still        1
20    listen        1
21    arrives,       1
22    progres       2
------------------------------------------------------
```

ASSIGNMENT NO.2

Title: Implement Single-pass Algorithm for clustering of files.

Code:

```
#include<stdio.h>
#include<conio.h>
#include<math.h>
/********************************************************************/
int no=0,i=0,k=0,j=0,flag=0;
char filename[10][20];
char kw[30][30],token[30];
int doc[150],dummy;
float centroid[150],temp=0,temp1=0,temp2=0;
float temp3[10];
char cl1[15][3];
FILE *fp;
/********************************************************************/
void keywords()
{
k=0;
for(i=1;i<=no;i++)
{
fp=fopen(filename[i],"r");
if(fp==NULL)
{
printf("\nCant read input file");
}
else
{
while(!feof(fp))
```

```
{
fscanf(fp,"%s",token);
fscanf(fp,"%d",&dummy);
flag=0;
for(j=0;j<k;j++)
{
if(strcmp(kw[j],token)==0)
{
flag=1;
break;
}}
if(flag==0)
{
strcpy(kw[k],token);
k++;
}}}
fclose(fp);
}
for(i=1;i<=no;i++)
{
fp=fopen(filename[i],"r");
if(fp!=NULL)
{
while(!feof(fp))
{
fscanf(fp,"%s",token);
fscanf(fp,"%d",&dummy);
for(j=0;j<k;j++)
{
if(strcmp(kw[j],token)==0)
{
doc[k*(i-1)+j]=dummy;
break;
}}}}
fclose(fp);
}}
/***********************************************************************/
void clustering()
{
int l=0,cluster,pri=0,m=0,c=0;
char ctemp[3]="C1",ctemp1[3]="D1";
float temp31;
for(i=0;i<k;i++)
{
centroid[i]=doc[i];
}
strcpy(cl1[0],ctemp);
strcpy(cl1[1],ctemp1);
printf("\n%s = { %s }",cl1[0],cl1[1]);
pri=2;
cluster=0;
for(j=1;j<no;j++)
{
m=0;
printf("\n\n\n\nConsider D%d,",j+1);
while(m<=cluster)
{
temp=0;temp1=0;temp2=0;
for(i=0;i<k;i++)
{
temp=temp+(centroid[m*k+i]*doc[j*k+i]);
temp1=temp1+(pow(centroid[m*k+i],2));
temp2=temp2+(pow(doc[j*k+i],2));
}
temp3[m]=(float)(2*temp)/(temp1+temp2);
printf("\n\nSim( D%d , C%d )= %.2f ",j+1,m+1,temp3[m]);
```

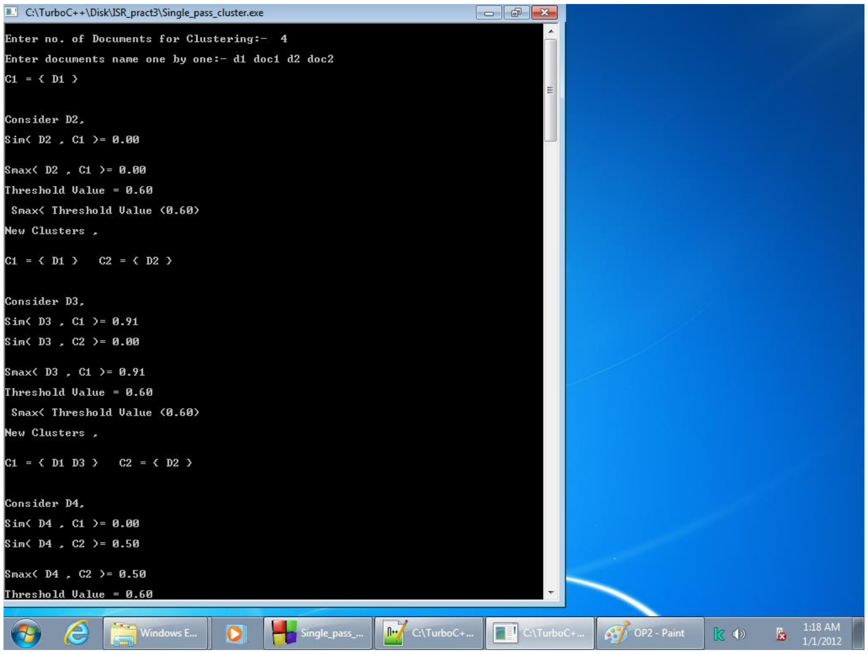
```
m++;
}
temp31=temp3[0];
c=0;
for(m=1;m<=cluster;m++)
{
if(temp31<temp3[m])
{
temp31=temp3[m];
c=m;
}}
if(temp31>0.60)
{
printf("\n\n\nSmax( D%d , C%d )= %.2f",j+1,c+1,temp31);
printf("\n\nThreshold Value = 0.60");
printf("\n\n Smax< Threshold Value (0.60)");
printf("\n\nNew Clusters ,");
if(c!=cluster)
{
for(m=0;strcmp(cl1[m],"\0")!=0;m++)
{
}
ctemp[1]=c+2+48;
for(i=m;strcmp(cl1[i],ctemp)!=0;i--)
{
strcpy(cl1[i],cl1[i-1]);
}
strcpy(cl1[i],cl1[i-1]);
ctemp1[1]=j+1+48;
strcpy(cl1[i],ctemp1);
pri++;
}
else
```

```
{
ctemp1[1]=j+1+48;
strcpy(cl1[pri++],ctemp1);
}
for(i=0;i<k;i++)
{
centroid[c*k+i]=(centroid[c*k+i]+doc[j*k+i])/2;
}}
else
{
printf("\n\n\nSmax( D%d , C%d )= %.2f",j+1,c+1,temp31);
printf("\n\nThreshold Value = 0.60");
printf("\n\n Smax< Threshold Value (0.60)");
printf("\n\nNew Clusters ,");
cluster++;
ctemp[1]=cluster+1+48;
strcpy(cl1[pri++],ctemp);
ctemp1[1]=j+1+48;
strcpy(cl1[pri++],ctemp1);
for(i=0;i<k;i++)
{
centroid[cluster*k+i]=doc[j*k+i];
}}
printf("\n\n\n");
for(i=0;strcmp(cl1[i],"\0")!=0;i++)
{
if(cl1[i][0]=='C')
{
printf("%s = { ",cl1[i]);
l=i+1;
while(cl1[l][0]!='C'&&strcmp(cl1[l],"\0")!=0)
{
printf("%s ",cl1[l]);
l++;
}
printf("}    ");
}}
getch();
}
}
/****************************************************************************/
void main()
{
//clrscr();
printf("\nEnter no. of Documents for Clustering:-  ");
scanf("%d",&no);
printf("\nEnter documents name one by one:- ");
for(i=1;i<=no;i++)
{
scanf("%s",filename[i]);
}
//clrscr();
for(i=1;i<=no;i++)
{
strcat(filename[i],".txt");
}
if(no<=1)
{
printf("\nNo. of documents should be greater than 1!!!!");
getch();
exit(0);
}
else
{
keywords();
clustering();
```

}}

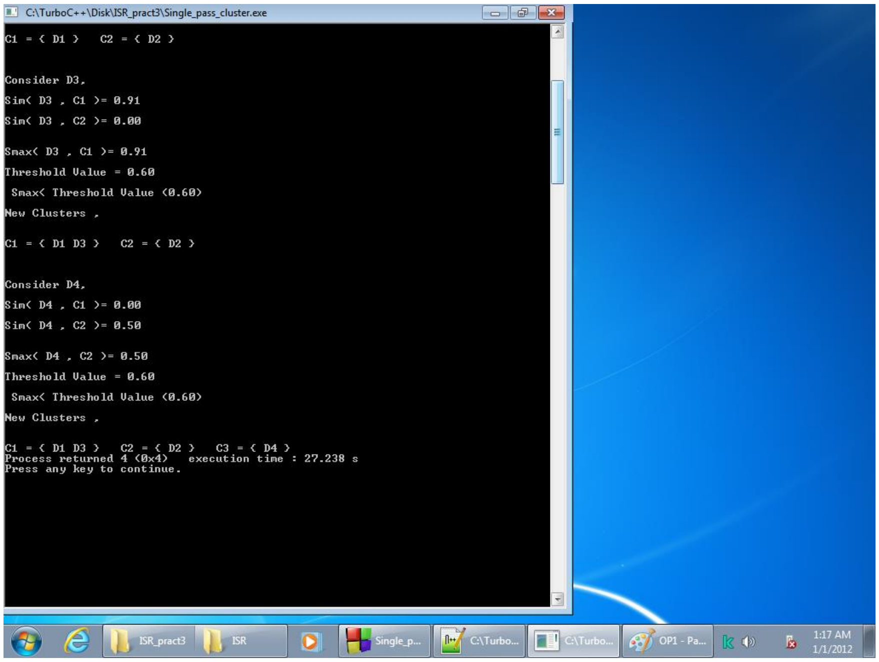
Input: Document (d1, d2)

Output:



C:\TurboC++\Disk\ISR_pract3\Single_pass_cluster.exe

```
Enter no. of Documents for Clustering:-  4
Enter documents name one by one:- d1 doc1 d2 doc2
C1 = { D1 }


Consider D2,
Sim( D2 , C1 )= 0.00

Smax( D2 , C1 )= 0.00
Threshold Value = 0.60
 Smax< Threshold Value (0.60)
New Clusters ,

C1 = { D1 }   C2 = { D2 }


Consider D3,
Sim( D3 , C1 )= 0.91
Sim( D3 , C2 )= 0.00

Smax( D3 , C1 )= 0.91
Threshold Value = 0.60
 Smax< Threshold Value (0.60)
New Clusters ,

C1 = { D1 D3 }   C2 = { D2 }


Consider D4,
Sim( D4 , C1 )= 0.00
Sim( D4 , C2 )= 0.50

Smax( D4 , C2 )= 0.50
Threshold Value = 0.60
```



C:\TurboC++\Disk\ISR_pract3\Single_pass_cluster.exe

```
C1 = { D1 }   C2 = { D2 }


Consider D3,
Sim( D3 , C1 )= 0.91
Sim( D3 , C2 )= 0.00

Smax( D3 , C1 )= 0.91
Threshold Value = 0.60
 Smax< Threshold Value (0.60)
New Clusters ,

C1 = { D1 D3 }   C2 = { D2 }


Consider D4,
Sim( D4 , C1 )= 0.00
Sim( D4 , C2 )= 0.50

Smax( D4 , C2 )= 0.50
Threshold Value = 0.60
 Smax< Threshold Value (0.60)
New Clusters ,

C1 = { D1 D3 }   C2 = { D2 }   C3 = { D4 }
Process returned 4 (0x4)   execution time : 27.238 s
Press any key to continue.
```

ASSIGNMENT NO3

Title: To implement a program for Retrieval of documents using inverted files.

Code:

Inverted.cpp

```
#include<iostream>
#include<vector>
#include<map>
#include<string>
#include<fstream>
#include<sstream>
using namespace std;

struct word_position
{
  string file_name;
  int line;
  int index;
};

class InvertedIndex
{
  map<string,vector<word_position> > Dictionary;
  vector<string> filelist;

  public:
    void addfile(string filename);
    void show_files();
    void search(string word);
};

void InvertedIndex::addfile(string filename)
```

```
{
  ifstream fp;
  fp.open(filename + ".txt",ios::in);

  if(!fp)
  {
    cout<<"File Not Found\n";
    return ;
  }

  filelist.push_back(filename);

  string line,word;
  int line_number=0,word_number=0;
  while(getline(fp,line))
  {
    line_number++;
    word_number = 0;
    stringstream s(line);
    while(s>>word)
    {
      word_number++;
      word_position obj;
      obj.file_name = filename;
      obj.line = line_number;
      obj.index = word_number;
      Dictionary[word].push_back(obj);
    }
  }
  fp.close();
}

void InvertedIndex::show_files()
```

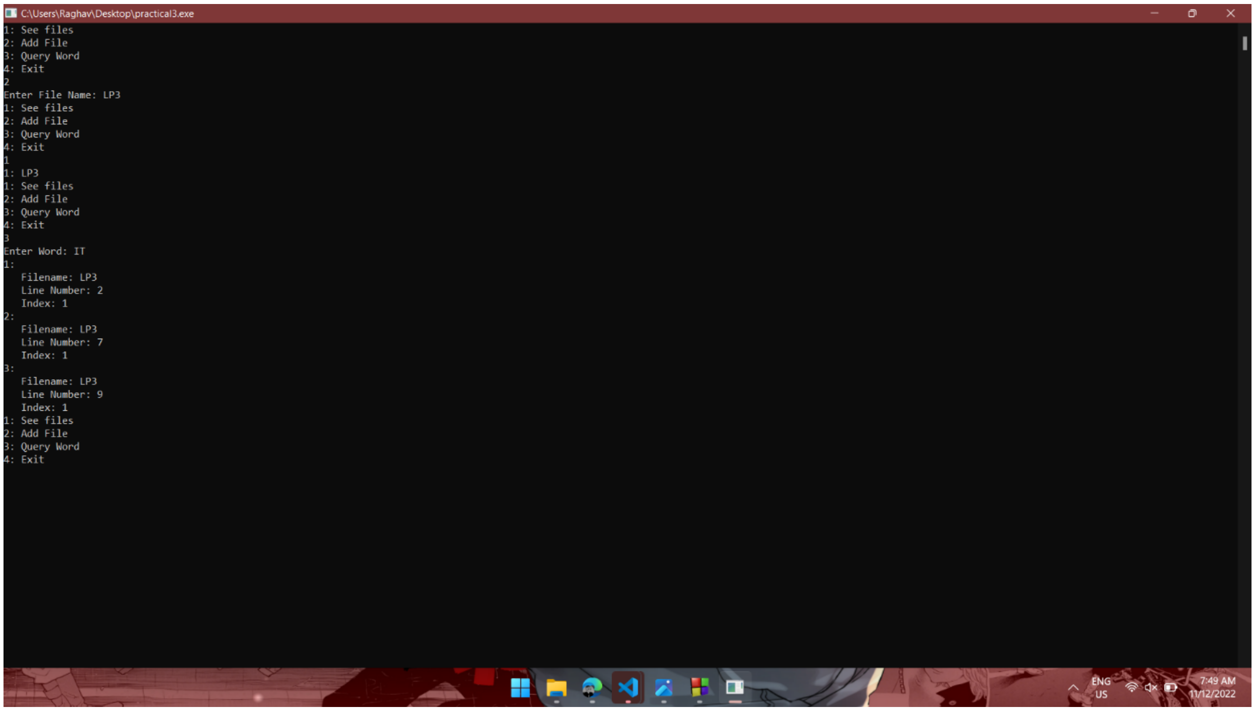
```
{
  int size = (int)filelist.size();
  for(int i=0;i<size;i++) cout<<i+1<<": "<<filelist[i]<<endl;

  if(!size) cout<<"No files added\n";
}

void InvertedIndex::search(string word)
{
  if(Dictionary.find(word)==Dictionary.end())
  {
    cout<<"No instance exist\n";
    return ;
  }



  int size = (int)Dictionary[word].size();
  for(int counter = 0;counter < size ;counter++)
  {
    cout<<counter+1<<":\n";
    cout<<"   Filename: "<<Dictionary[word][counter].file_name<<endl;
    cout<<"   Line Number: "<<Dictionary[word][counter].line<<endl;
    cout<<"   Index: "<<Dictionary[word][counter].index<<endl;
  }
}

int main(int argc, char*argv[])
{
  InvertedIndex Data;
  for(int i = 1 ; i< argc ; i++)
  {
    Data.addfile(argv[i]);
  }
```

```
int choice = 0;
do
{
  cout<<"1: See files\n2: Add File\n3: Query Word\n4: Exit\n";
  cin>>choice;
  switch(choice)
  {
    case 1: Data.show_files(); break;
    case 2:
    {
      cout<<"Enter File Name: ";
      string name;
      cin>>name;
      Data.addfile(name);
      break;
    }

    case 3:
    {
      cout<<"Enter Word: ";
      string word;
      cin>>word;
      Data.search(word);
      break;
    }

    case 4: break;

    default : continue;
  }
}while(choice!=4);
```

```
  return 0;
}
```

Input: Lp3.txt

Output:



```
1: See files
2: Add File
3: Query Word
4: Exit
2
Enter File Name: LP3
1: See files
2: Add File
3: Query Word
4: Exit
1
1: LP3
1: See files
2: Add File
3: Query Word
4: Exit
3
Enter Word: IT
1:
   Filename: LP3
   Line Number: 2
   Index: 1
2:
   Filename: LP3
   Line Number: 7
   Index: 1
3:
   Filename: LP3
   Line Number: 9
   Index: 1
1: See files
2: Add File
3: Query Word
4: Exit
```

Assignment No.4 & 5

Title (4th): Implement a program to calculate precision and recall for sample input.

Title (5th): Write a program to calculate harmonic mean (F-measure) and E-measure for above example.

Code:

```
#include <iostream>

#include <string.h>

#include <iomanip>

#include <fstream>


using namespace std;


string left(const string s, const int w)

{ // Left aligns input string in table

   stringstream ss, spaces;

   int padding = w - s.size(); // count excess room to pad

   for (int i = 0; i < padding;  ++i)

      spaces << " ";

   ss << s << spaces.str() << '|'; // format with padding

   return ss.str();

}


string center(const string s, const int w)

{ // center aligns input string in table

   stringstream ss, spaces;

   int padding = w - s.size(); // count excess room to pad

   for (int i = 0; i < padding / 2; ++i)

      spaces << " ";

   ss << spaces.str() << s << spaces.str(); // format with padding

   if (padding > 0 && padding % 2 != 0)     // if odd #, add 1 space
```

```
        ss << " ";
    return ss.str();
}

string prd(float x, int decDigits, int width)
{ // right aligns float values with specified no. of precision digits in a table
    stringstream ss;
    ss << fixed << right;
    ss.fill(' ');            // fill space around displayed #
    ss.width(width);         // set  width around displayed #
    ss.precision(decDigits); // set # places after decimal
    ss << x;
    return ss.str();
}

string printDocs(string state[], int size)
{
    // prints each document at a specific iteration inside the table
    stringstream ss;
    ss << '|' << ' ';
    for (int i = 0; i < size; i++)
    { // convert the array into a string of comma seprated values
        ss << state[i];
        if (state[i].compare("") != 0 and i + 1 < size and state[i + 1].compare("") != 0)
            ss << ',' << ' ';
    }
    return left(ss.str(), 98);
}
float E_value(float b, float rj, float pj)
{ // calculates E value
    return 1 - (((1 + b * b) * rj * pj) / (b * b * pj + rj));
}
```

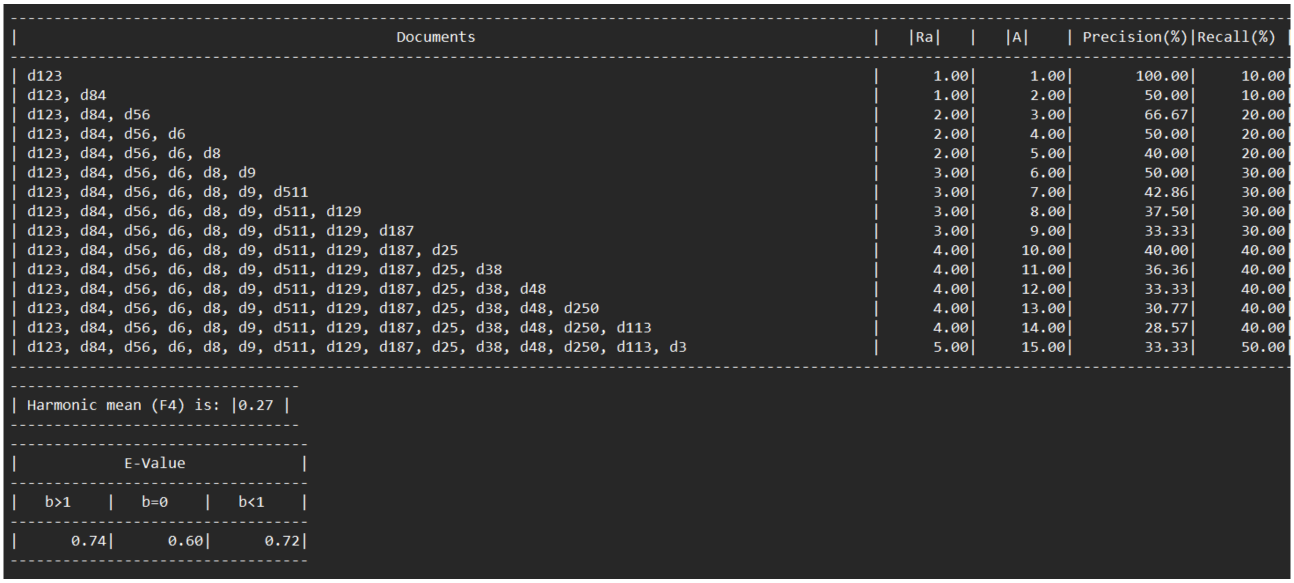
```
int main()
{ // Hardcoded Rq and A
    string Rq[10] = {"d3", "d5", "d9", "d25", "d39", "d44", "d56", "d71", "d89", "d123"};
    string A[15] = {"d123", "d84", "d56", "d6", "d8", "d9", "d511", "d129", "d187", "d25", "d38", "d48", "d250",
"d113", "d3"};


    // Creating and opening output file
    ofstream write("Recall_Precision_Evaluation_output.txt");


    // required constants and arrays for calculations
    float modRq = sizeof(Rq) / sizeof(Rq[0]);
    string Ra[sizeof(A) / sizeof(A[0])];
    float P[sizeof(A) / sizeof(A[0])];
    float R[sizeof(A) / sizeof(A[0])];
    float modRa = 0;
    float modA = 0;
    double precision;
    double recall;


    // table header formatting and printing
    std::cout << setprecision(2) << fixed;
    write << setprecision(2) << fixed;
    std::cout << string(45 * 3 + 11, '-') << "\n";
    write << string(45 * 3 + 11, '-') << "\n";
    std::cout << '|' << center("Documents", 96) << " | "
              << center("|Ra|", 8) << " | "
              << center("|A|", 8) << " | "
              << center("Precision(%)", 5) << "|"
              << center("Recall(%)", 5) << " | " << endl;
    write << '|' << center("Documents", 96) << " | "
          << center("|Ra|", 8) << " | "
          << center("|A|", 8) << " | "
          << center("Precision(%)", 5) << "|"
          << center("Recall(%)", 5) << " | " << endl;
```

```
std::cout << string(45 * 3 + 11, '-') << "\n";
write << string(45 * 3 + 11, '-') << "\n";
// Algorithm to calculate and print all the values in the output table, MAIN algo
for (int i = 0; i < sizeof(A) / sizeof(A[0]); i++)
{
    Ra[i] = A[i];
    modA++;
    for (int j = 0; j < modRq; j++)
    {
        if (A[i] == Rq[j])
        {
            modRa++;
            break;
        }
    }
    precision = (modRa / modA) * 100;
    P[i] = precision / 100;
    recall = (modRa / modRq) * 100;
    R[i] = recall / 100;
    // Printing documents and other values of current iteration within the table
    std::cout << printDocs(Ra, sizeof(Ra) / sizeof(Ra[0]));
    write << printDocs(Ra, sizeof(Ra) / sizeof(Ra[0]));
    std::cout << prd(modRa, 2, 10) << "|"
              << prd(modA, 2, 10) << "|"
              << prd(precision, 2, 13) << "|"
              << prd(recall, 2, 10) << "|"
              << endl;
    write << prd(modRa, 2, 10) << "|"
          << prd(modA, 2, 10) << "|"
          << prd(precision, 2, 13) << "|"
          << prd(recall, 2, 10) << "|"
          << endl;
}
```

```
    // closing the table
    std::cout << string(45 * 3 + 11, '-') << "\n";
    write << string(45 * 3 + 11, '-') << "\n";

    // taking user input for calculation of Fj and Ej
    int j;
    do
    {
        std::cout << "Harmonic mean and E-value\nEnter value of j(0 - " << (sizeof(A) / sizeof(A[0])) - 1 << ") to find F(j)and E(j):" << endl;
        cin >> j;
    } while (j > sizeof(Ra) / sizeof(Ra[0]));

    // calculating Harmonic mean and printing in table
    float Fj = (2 * P[j] * R[j]) / (P[j] + R[j]);
    std::cout << string(15 * 2 + 3, '-') << "\n"
              << "| Harmonic mean (F" << j << ") is: |" << Fj << " |\n"
              << string(15 * 2 + 3, '-') << "\n";
    write << string(15 * 2 + 3, '-') << "\n"
          << "| Harmonic mean (F" << j << ") is: |" << Fj << " |\n"
          << string(15 * 2 + 3, '-') << "\n";

    // table header
    std::cout << string(15 * 2 + 4, '-') << "\n"
              << "|" << center("E-Value", 32) << "|\n"
              << string(15 * 2 + 4, '-') << "\n";
    write << string(15 * 2 + 4, '-') << "\n"
          << "|" << center("E-Value", 32) << "|\n"
          << string(15 * 2 + 4, '-') << "\n";

    // table header (sub columns)
    std::cout << "|" << center("b>1", 10) << "|"
              << center("b=0", 10) << "|"
              << center("b<1", 10) << "|\n"
```

```
            << string(15 * 2 + 4, '-') << "\n";
    write << "|"  << center("b>1", 10) << "|"
          << center("b=0", 10) << "|"
          << center("b<1", 10) << "|\n"
          << string(15 * 2 + 4, '-') << "\n";

    // Calculating and Printing E-Values in table
    std::cout << "|" << prd(E_value(1.1, R[j], P[j]), 2, 10) << "|"
            << prd(E_value(0, R[j], P[j]), 2, 10) << "|"
            << prd(E_value(0.9, R[j], P[j]), 2, 10) << "|\n";
    write << "|"  << prd(E_value(1.1, R[j], P[j]), 2, 10) << "|"
          << prd(E_value(0, R[j], P[j]), 2, 10) << "|"
          << prd(E_value(0.9, R[j], P[j]), 2, 10) << "|\n";
    // Closing table
    std::cout << string(15 * 2 + 4, '-') << "\n";
    write << string(15 * 2 + 4, '-') << "\n";
    write.close();
    return 0;
}
```

Output:



```
--------------------------------------------------------------------------------------------------------------------------------------------------
|                                           Documents                                             |   |Ra|    |    |A|    | Precision(%)|Recall(%) |
--------------------------------------------------------------------------------------------------------------------------------------------------
| d123                                                                                            |      1.00|       1.00|        100.00|     10.00|
| d123, d84                                                                                       |      1.00|       2.00|         50.00|     10.00|
| d123, d84, d56                                                                                  |      2.00|       3.00|         66.67|     20.00|
| d123, d84, d56, d6                                                                              |      2.00|       4.00|         50.00|     20.00|
| d123, d84, d56, d6, d8                                                                          |      2.00|       5.00|         40.00|     20.00|
| d123, d84, d56, d6, d8, d9                                                                      |      3.00|       6.00|         50.00|     30.00|
| d123, d84, d56, d6, d8, d9, d511                                                                |      3.00|       7.00|         42.86|     30.00|
| d123, d84, d56, d6, d8, d9, d511, d129                                                          |      3.00|       8.00|         37.50|     30.00|
| d123, d84, d56, d6, d8, d9, d511, d129, d187                                                    |      3.00|       9.00|         33.33|     30.00|
| d123, d84, d56, d6, d8, d9, d511, d129, d187, d25                                               |      4.00|      10.00|         40.00|     40.00|
| d123, d84, d56, d6, d8, d9, d511, d129, d187, d25, d38                                          |      4.00|      11.00|         36.36|     40.00|
| d123, d84, d56, d6, d8, d9, d511, d129, d187, d25, d38, d48                                     |      4.00|      12.00|         33.33|     40.00|
| d123, d84, d56, d6, d8, d9, d511, d129, d187, d25, d38, d48, d250                               |      4.00|      13.00|         30.77|     40.00|
| d123, d84, d56, d6, d8, d9, d511, d129, d187, d25, d38, d48, d250, d113                         |      4.00|      14.00|         28.57|     40.00|
| d123, d84, d56, d6, d8, d9, d511, d129, d187, d25, d38, d48, d250, d113, d3                     |      5.00|      15.00|         33.33|     50.00|
--------------------------------------------------------------------------------------------------------------------------------------------------
---------------------------------
| Harmonic mean (F4) is: |0.27 |
---------------------------------
----------------------------------
|            E-Value             |
----------------------------------
|   b>1    |   b=0    |   b<1    |
----------------------------------
|      0.74|      0.60|      0.72|
----------------------------------
```

Assignment No. 6

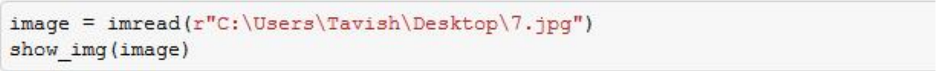
Problem Statement:

To Implement a program for feature extraction in 2D color images (any features like color, texture etc. and to extract features from input image and plot histogram for the features.

Steps:-

A. Importing an Image:

Importing an image in python is easy. Following code will help you import an image on Python :

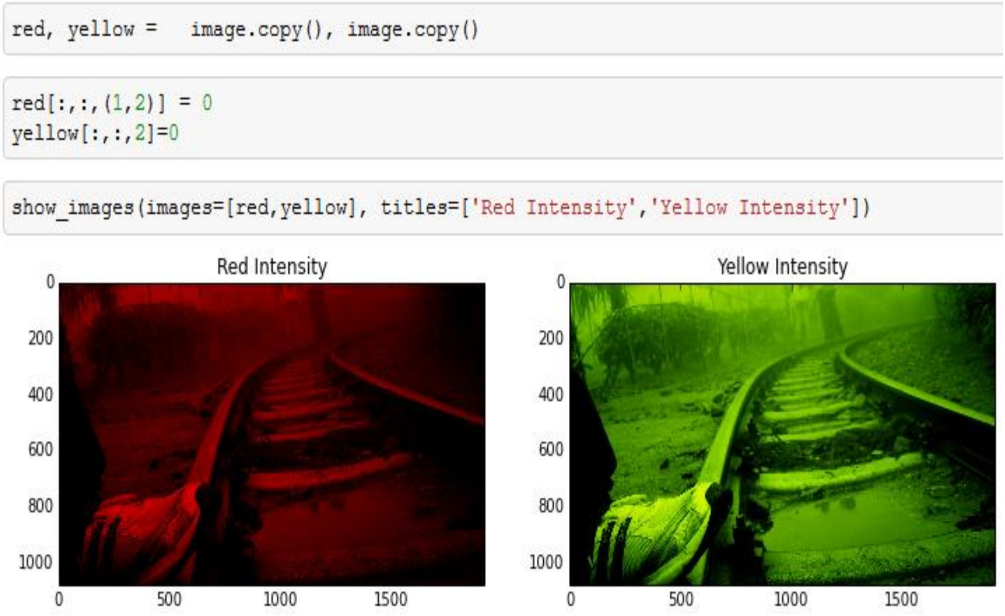


```
image = imread(r"C:\Users\Tavish\Desktop\7.jpg")
show_img(image)
```

B. Understanding the underlying data:

This image has several colors and many pixels.

1. To visualize how this image is stored, think of every pixel as a cell in matrix.
2. Now this cell contains three different intensity information, catering to the color Red, Green and Blue. So a RGB image becomes a 3-D matrix.
3. Each number is the intensity of Red, Blue and Green colors.



```
red, yellow =   image.copy(), image.copy()
```

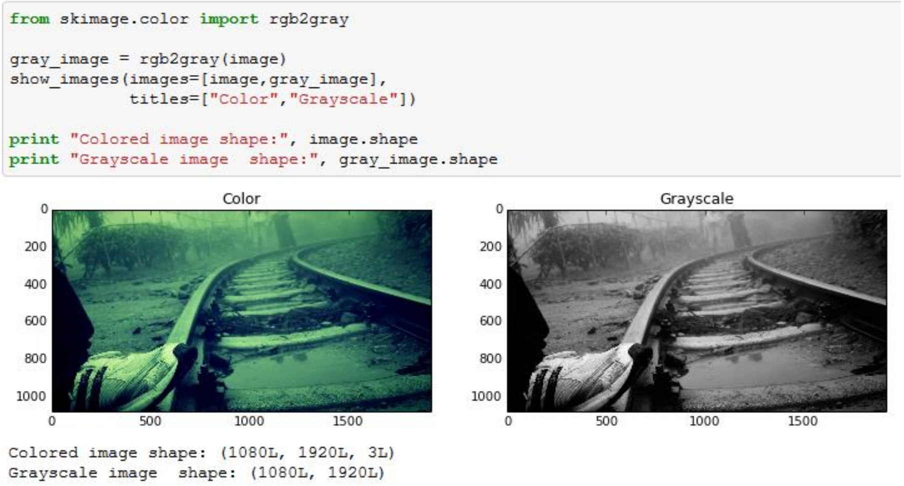
```
red[:,:,(1,2)] = 0
yellow[:,:,2]=0
```

```
show_images(images=[red,yellow], titles=['Red Intensity','Yellow Intensity'])
```

C. Converting Images to a 2-D matrix:-

1. Handling the third dimension of images sometimes can be complex and redundant.
2. In feature extraction, it becomes much simpler if we compress the image to a 2-D matrix.

3. This is done by Gray-scaling ,Here is how you convert a RGB image to Gray scale.



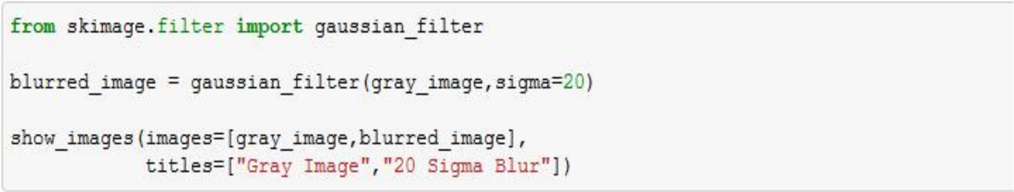
```
from skimage.color import rgb2gray

gray_image = rgb2gray(image)
show_images(images=[image,gray_image],
            titles=["Color","Grayscale"])

print "Colored image shape:", image.shape
print "Grayscale image  shape:", gray_image.shape
```

```
Colored image shape: (1080L, 1920L, 3L)
Grayscale image  shape: (1080L, 1920L)
```

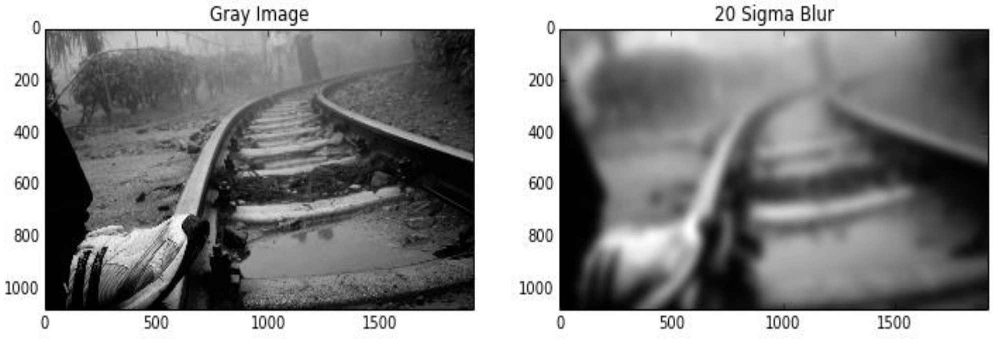
Now let's try to binarize this Gray scale image:-

Blurring an Image:-

Last part of this assignment is more relevant for feature extraction : Blurring of images.



```
from skimage.filter import gaussian_filter

blurred_image = gaussian_filter(gray_image,sigma=20)

show_images(images=[gray_image,blurred_image],
            titles=["Gray Image","20 Sigma Blur"])
```



Example:-

```
image = imread(r"C:\Users\Tavish\Desktop\7.jpg")
show_img(image)
red, yellow =  image.copy(), image.copy()
red[:,:,(1,2)] = 0
yellow[:,:,2]=0
show_images(images=[red,yellow], titles=['Red Intensity','Yellow Intensity'])
 from skimage.color  import rgb2gray
gray_image = rgb2gray(image)
show_images(images=[image,gray_image],titles=["Color","Grayscale"])



print "Colored image shape:", image.shape.
print "Grayscale image shape:", gray_image.shape
from skimage.filter
import threshold_otsu
thresh = threshold_otsu(gray_image)
binary = gray_image > thresh
```

```
show_images(images=[gray_image,binary_image,binary],titles=["Grayscale" ,"Otsu Binary"])
from skimage.filter import gaussian_filter
blurred_image = gaussian_filter(gray_image,sigma=20)
show_images(images=[gray_image,blurred_image],titles=["Gray Image","20 Sigma Blur"])
```

Second Part of the Assignment –Plotting the Histogram

histogram is a graphical representation showing how frequently various colour values occur in the image.

Steps:-

Importing image data:-

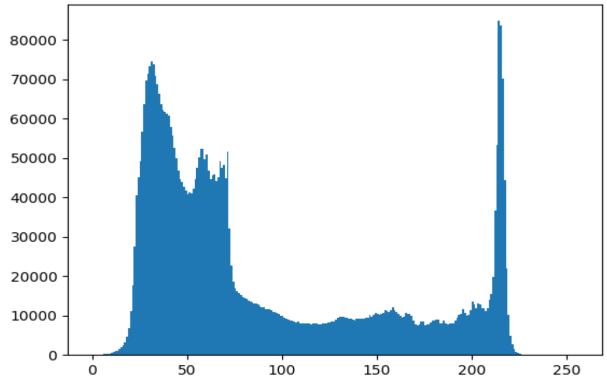
```
import matplotlib.pyplot as plt          #importing matplotlib
```

The image should be used in a PNG file as matplotlib supports only PNG images.

```
img = plt.imread('flower.png')          #reads image data
```

Histogram creation using numpy array:-

- To create a histogram of our image data, we use the hist() function.
- plt.hist(n_img.ravel(), bins=256, range=(0.0, 1.0), fc='k', ec='k') #calculating histogram



Histogram Calculation:-

- Here, we use cv2.calcHist()(in-built function in OpenCV) to find the histogram.
- cv2.calcHist(images, channels, mask, histSize, ranges[, hist[, accumulate]])

images : it is the source image of type uint8 or float32 represented as "[img]".

channels : it is the index of channel for which we calculate histogram.

For grayscale image, its value is [0] and color image, you can pass [0], [1] or [2] to calculate histogram of blue, green or red channel respectively.

mask : mask image. To find histogram of full image, it is given as "None".

histSize : this represents our BIN count. For full scale, we pass [256].

ranges : this is our RANGE. Normally, it is [0,256].

Example:-

```
# load an image in grayscale mode
img = cv2.imread('ex.jpg',0)
# calculate frequency of pixels in range 0-255
histg = cv2.calcHist([img],[0],None,[256],[0,256])
```

Then, we need to plot histogram to show the characteristics of an image.

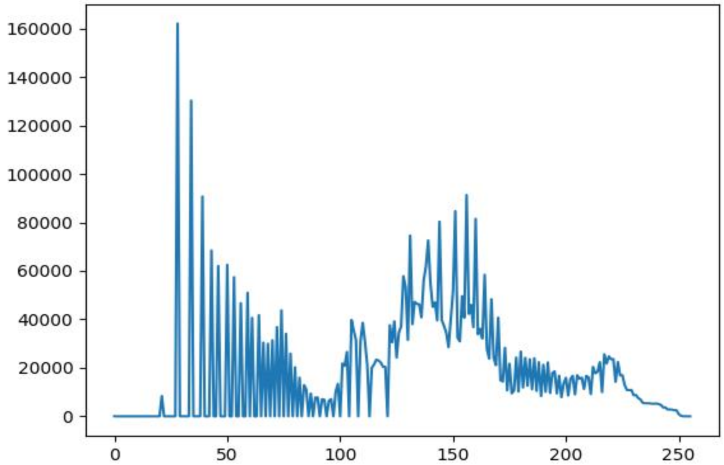
Plotting Histograms

Analysis using Matplotlib:

```
# importing required libraries of opencv
import cv2



# importing library for plotting
from matplotlib import pyplot as plt
# reads an input image
img = cv2.imread('ex.jpg',0)
# find frequency of pixels in range 0-255
histr = cv2.calcHist([img],[0],None,[256],[0,256])
# show the plotting graph of an image
plt.plot(histr)
plt.show()
```

Input:

Output:


160000
140000
120000
100000
80000
60000
40000
20000
0
0
50
100
150
200
250

ASSIGNMENT NO.7

Title: Build the web crawler to pull product information and links from an e-commerce website. (Python)

Input: product Url link

Explanation:

Searchresults.py contains the python code for execution.

search_results.yml contains the format in which the data is going to be extracted.

search_results_urls.txt contains the url from which the product information is going to be retrieved

search_results_output.jsonl contains the output generated i.e. the products information.

Code:

Searchresults.py

```
from selectorlib import Extractor
import requests
import json
from time import sleep
# Create an Extractor by reading from the YAML file
e = Extractor.from_yaml_file('search_results.yml')

def scrape(url):
    headers = {
        'dnt': '1',
        'upgrade-insecure-requests': '1',
        'user-agent': 'Mozilla/5.0 (Macintosh; Intel Mac OS X 10_15_4) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/83.0.4103.61 Safari/537.36',
        'accept': 'text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.9',
        'sec-fetch-site': 'same-origin',
        'sec-fetch-mode': 'navigate',
        'sec-fetch-user': '?1',
        'sec-fetch-dest': 'document',
        'referer': 'https://www.amazon.com/',
        'accept-language': 'en-GB,en-US;q=0.9,en;q=0.8',
    }

    # Download the page using requests
    print("Downloading %s"%url)
    r = requests.get(url, headers=headers)
    # Simple check to check if page was blocked (Usually 503)
    if r.status_code > 500:
        if "To discuss automated access to Amazon data please contact" in r.text:
            print("Page %s was blocked by Amazon. Please try using better proxies\n"%url)
        else:
            print("Page %s must have been blocked by Amazon as the status code was %d"%(url,r.status_code))
        return None
    # Pass the HTML of the page and create
    return e.extract(r.text)
# product_data = []
with open("search_results_urls.txt",'r') as urllist, open('search_results_output.jsonl','w') as outfile:
    for url in urllist.read().splitlines():
        data = scrape(url)
        if data:
            for product in data['products']:
                product['search_url'] = url
                print("Saving Product: %s"%product['title'])
                json.dump(product,outfile)
                outfile.write("\n")
                # sleep(5)
```

search_results.yml

```
products:
    css: 'div[data-component-type="s-search-result"]'
    xpath: null
    multiple: true
    type: Text
    children:
        title:
            css: 'h2 a.a-link-normal.a-text-normal'
            xpath: null
            type: Text
        url:
            css: 'h2 a.a-link-normal.a-text-normal'
            xpath: null
            type: Link
        rating:
            css: 'div.a-row.a-size-small span:nth-of-type(1)'
            xpath: null
            type: Attribute
            attribute: aria-label
        reviews:
            css: 'div.a-row.a-size-small span:nth-of-type(2)'
            xpath: null
            type: Attribute
            attribute: aria-label
        price:
            css: 'span.a-price:nth-of-type(1) span.a-offscreen'
            xpath: null
            type: Text
```

search_results_urls.txt

https://www.amazon.com/s?k=iphones

search_results_output.jsonl

{"title": "Apple iPhone 13 Pro Max, 128GB, Sierra Blue - Unlocked (Renewed)", "url": "/Apple-iPhone-13-Pro-Max/dp/B09LPDM924/ref=sr_1_1?keywords=iphone&qid=1667294247&qu=eyJxc2MiOiI3Ljk2IiwicXNhIjoiNy42MyIsInFzcC

I6IjcuMTEifQ%3D%3D&sr=8-1", "rating": "4.3 out of 5 stars", "reviews": "363", "price": "$889.00", "search_url": "https://www.amazon.com/s?k=iphones"}

{"title": "Apple iPhone X, 64GB, Space Gray - Fully Unlocked (Renewed Premium)", "url": "/Apple-iPhone-64GB-Space-Gray/dp/B08BDTL7NP/ref=sr_1_2?keywords=iphone&qid=1667294247&qu=eyJxc2MiOiI3Ljk2IiwicXNhIjoiNy42MyIsInFzcCI6IjcuMTEifQ%3D%3D&sr=8-2", "rating": "4.2 out of 5 stars", "reviews": "21,695", "price": "$299.00", "search_url": "https://www.amazon.com/s?k=iphones"}

{"title": "Apple iPhone SE, 128GB, Red - Fully Unlocked (Renewed Premium)", "url": "/Apple-iPhone-2nd-Generation-128GB/dp/B08R97FHJ7/ref=sr_1_3?keywords=iphone&qid=1667294247&qu=eyJxc2MiOiI3Ljk2IiwicXNhIjoiNy42MyIsInFzcCI6IjcuMTEifQ%3D%3D&sr=8-3", "rating": "4.4 out of 5 stars", "reviews": "6,884", "price": "$319.00", "search_url": "https://www.amazon.com/s?k=iphones"}

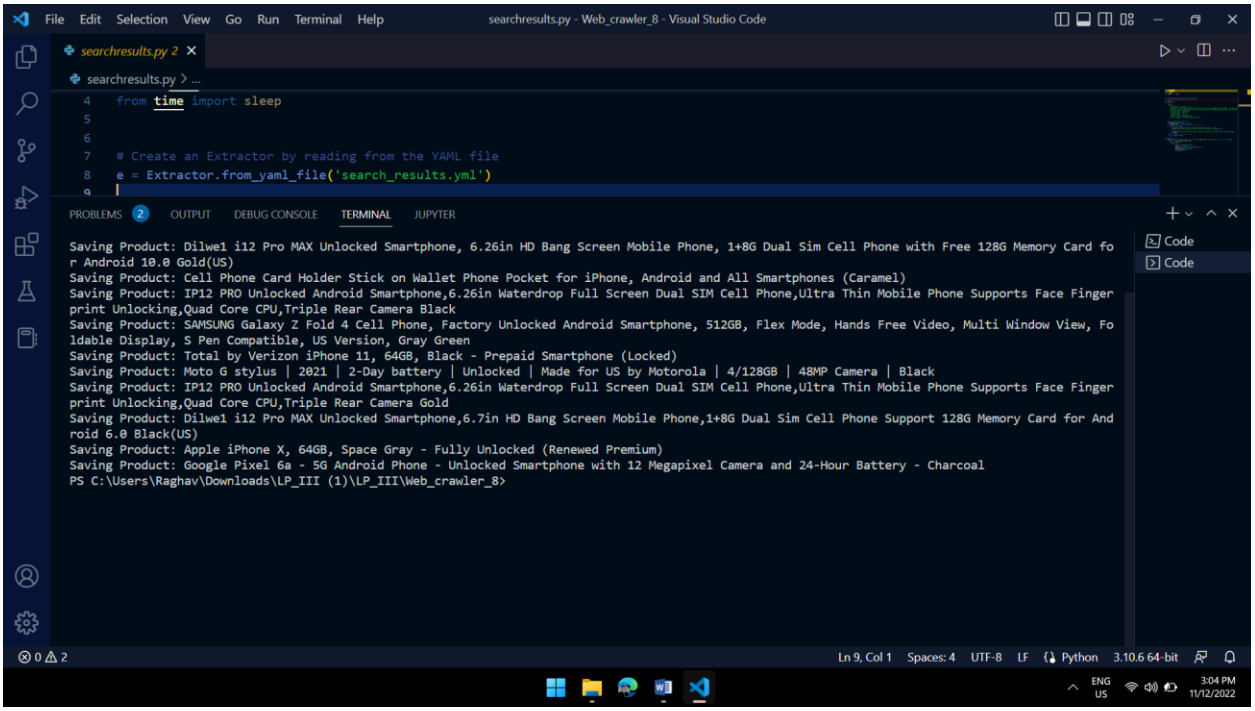
{"title": "Total by Verizon iPhone 11, 64GB, Black - Prepaid Smartphone (Locked)", "url": "/Total-Verizon-iPhone-11-Black/dp/B0BBXLNQ1R/ref=sr_1_4?keywords=iphone&qid=1667294247&qu=eyJxc2MiOiI3Ljk2IiwicXNhIjoiNy42MyIsInFzcCI6IjcuMTEifQ%3D%3D&sr=8-4", "rating": null, "reviews": null, "price": "$299.00", "search_url": "https://www.amazon.com/s?k=iphones"}

{"title": "Apple iPhone XR, US Version, 128GB, Red - Unlocked (Renewed)", "url": "/Apple-iPhone-Fully-Unlocked-128/dp/B07P611Q4N/ref=sr_1_5?keywords=iphone&qid=1667294247&qu=eyJxc2MiOiI3Ljk2IiwicXNhIjoiNy42MyIsInFzcCI6IjcuMTEifQ%3D%3D&sr=8-5", "rating": "4.5 out of 5 stars", "reviews": "53,432", "price": "$289.99", "search_url": "https://www.amazon.com/s?k=iphones"}

{"title": "Apple iPhone 6 16GB Factory Unlocked (ATT Tmobile Mint Ting Tello) Gold", "url": "/Apple-iPhone-16GB-Factory-Unlocked/dp/B08H72HZ2M/ref=sr_1_6?keywords=iphone&qid=1667294247&qu=eyJxc2MiOiI3Ljk2IiwicXNhIjoiNy42MyIsInFzcCI6IjcuMTEifQ%3D%3D&sr=8-6", "rating": "4.0 out of 5 stars", "reviews": "182", "price": null, "search_url": "https://www.amazon.com/s?k=iphones"}

{"title": "SAMSUNG Galaxy Z Fold 4 Cell Phone, Factory Unlocked Android Smartphone, 512GB, Flex Mode, Hands Free Video, Multi Window View, Foldable Display, S Pen Compatible, US Version, Gray Green", "url": "/SAMSUNG-Unlocked-Smartphone-Foldable-Compatible/dp/B0B4FBKLWP/ref=sr_1_7?keywords=iphone&qid=1667294247&qu=eyJxc2MiOiI3Ljk2IiwicXNhIjoiNy42MyIsInFzcCI6IjcuMTEifQ%3D%3D&sr=8-7", "rating": "4.4 out of 5 stars", "reviews": "11", "price": "$1,569.99", "search_url": "https://www.amazon.com/s?k=iphones"}

Output:



```
from time import sleep


# Create an Extractor by reading from the YAML file
e = Extractor.from_yaml_file('search_results.yml')
```

```
Saving Product: Dilwe1 i12 Pro MAX Unlocked Smartphone, 6.26in HD Bang Screen Mobile Phone, 1+8G Dual Sim Cell Phone with Free 128G Memory Card fo
r Android 10.0 Gold(US)
Saving Product: Cell Phone Card Holder Stick on Wallet Phone Pocket for iPhone, Android and All Smartphones (Caramel)
Saving Product: IP12 PRO Unlocked Android Smartphone,6.26in Waterdrop Full Screen Dual SIM Cell Phone,Ultra Thin Mobile Phone Supports Face Finger
print Unlocking,Quad Core CPU,Triple Rear Camera Black
Saving Product: SAMSUNG Galaxy Z Fold 4 Cell Phone, Factory Unlocked Android Smartphone, 512GB, Flex Mode, Hands Free Video, Multi Window View, Fo
ldable Display, S Pen Compatible, US Version, Gray Green
Saving Product: Total by Verizon iPhone 11, 64GB, Black - Prepaid Smartphone (Locked)
Saving Product: Moto G stylus | 2021 | 2-Day battery | Unlocked | Made for US by Motorola | 4/128GB | 48MP Camera | Black
Saving Product: IP12 PRO Unlocked Android Smartphone,6.26in Waterdrop Full Screen Dual SIM Cell Phone,Ultra Thin Mobile Phone Supports Face Finger
print Unlocking,Quad Core CPU,Triple Rear Camera Gold
Saving Product: Dilwe1 i12 Pro MAX Unlocked Smartphone,6.7in HD Bang Screen Mobile Phone,1+8G Dual Sim Cell Phone Support 128G Memory Card for And
roid 6.0 Black(US)
Saving Product: Apple iPhone X, 64GB, Space Gray - Fully Unlocked (Renewed Premium)
Saving Product: Google Pixel 6a - 5G Android Phone - Unlocked Smartphone with 12 Megapixel Camera and 24-Hour Battery - Charcoal
PS C:\Users\Raghav\Downloads\LP_III (1)\LP_III\Web_crawler_8>
```

ASSIGNMENT NO. 8

Title: Write a program to find the live weather report (temperature, wind speed, description, and weather) of a given city. (Python).

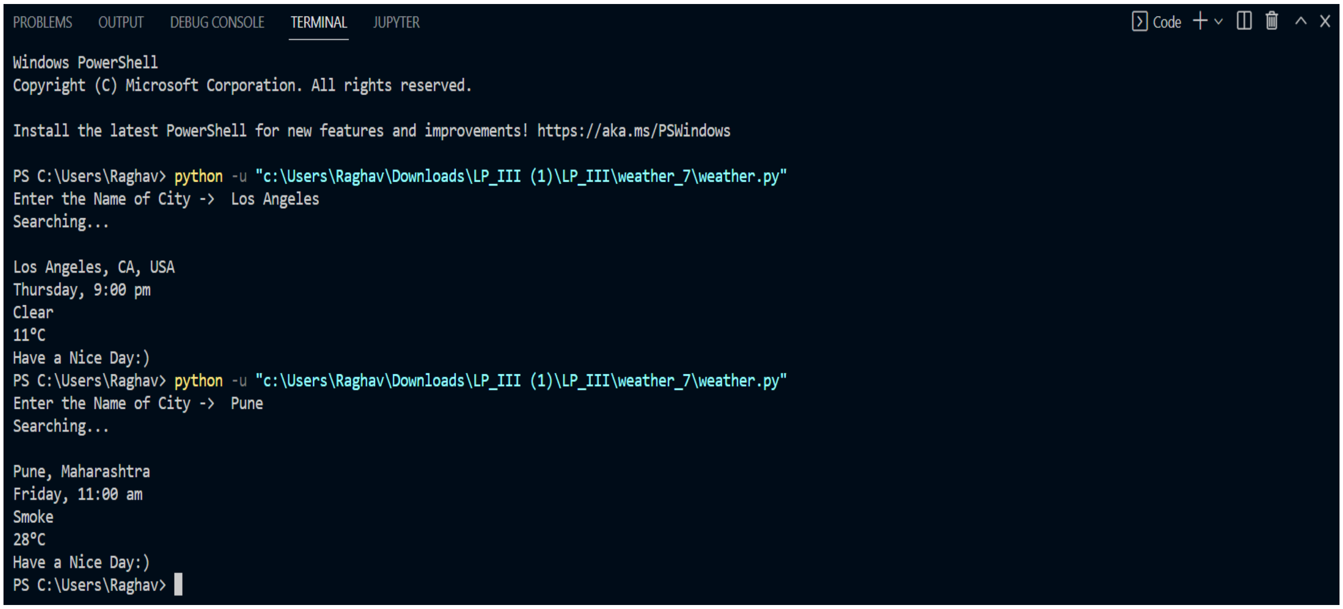
Input: City name

Code:

Weather.py

```
from bs4 import BeautifulSoup
import requests
headers = {
    'User-Agent': 'Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/58.0.3029.110 Safari/537.3'}
def weather(city):
    city = city.replace(" ", "+")
    res = requests.get(
        f'https://www.google.com/search?q={city}&oq={city}&aqs=chrome.0.35i39l2j0l4j46j69i60.6128j1j7&sourceid=chrome&ie=UTF-8', headers=headers)
    print("Searching...\n")
    soup = BeautifulSoup(res.text, 'html.parser')
    location = soup.select('#wob_loc')[0].getText().strip()
    time = soup.select('#wob_dts')[0].getText().strip()
    info = soup.select('#wob_dc')[0].getText().strip()
    weather = soup.select('#wob_tm')[0].getText().strip()
    print(location)
    print(time)
    print(info)
    print(weather+"°C")

city = input("Enter the Name of City ->  ")
city = city+" weather"
weather(city)
print("Have a Nice Day:)")
```

Output:



```
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Install the latest PowerShell for new features and improvements! https://aka.ms/PSWindows

PS C:\Users\Raghav> python -u "c:\Users\Raghav\Downloads\LP_III (1)\LP_III\weather_7\weather.py"
Enter the Name of City ->  Los Angeles
Searching...

Los Angeles, CA, USA
Thursday, 9:00 pm
Clear
11°C
Have a Nice Day:)
PS C:\Users\Raghav> python -u "c:\Users\Raghav\Downloads\LP_III (1)\LP_III\weather_7\weather.py"
Enter the Name of City ->  Pune
Searching...

Pune, Maharashtra
Friday, 11:00 am
Smoke
28°C
Have a Nice Day:)
PS C:\Users\Raghav>
```