**Task 1 (checking off)**

Pseudocode for the problem is given below

* READ, two strings s1 and s2
* LET, size1 = length of s1 and size2 = length of s2 and i = 0 and j = 0
* IF, size1 = size2 THEN
  + WHILE, I <= size1
    - WHILE, j <= size2
      * IF, s1[i] = s2[j], THEN
        + END WHILE
        + END IF
      * IF, j = size2, THEN
        + LET, check = false
        + END WHILE
        + END WHILE
        + END IF
        + END IF
      * LET, j++
      * END WHILE
    - LET, i++
    - END WHILE
  + LET, check = true
  + END IF
* ELSE
  + LET, check = false
* IF, check = true
  + PRINT, this is an anagram
  + END IF
* ELSE
  + PRINT, this is not an anagram

Program in c++

#include<iostream>

#include<string>

using namespace std;

bool anagram(char\*, char\*);

int main()

{

char str1[100], str2[100];

cout << "enter two string to check anagram" << endl;

cin >> str1 >> str2;

bool check = anagram(str1, str2);

if (check == true)

cout << "this is an anagram" << endl;

else

cout << "this is not an anagram" << endl;

system("pause");

}

bool anagram(char\*str1, char\*str2)

{

int size1 = strlen(str1), size2 = strlen(str2),i,j;

if (size1 == size2)

{

for (i = 0; i <= size1; i++)

{

for (j = 0; j <= size2; j++)

{

if (str1[i] == str2[j])

break;

if (j==size2)

return false;

}

}

return true;

}

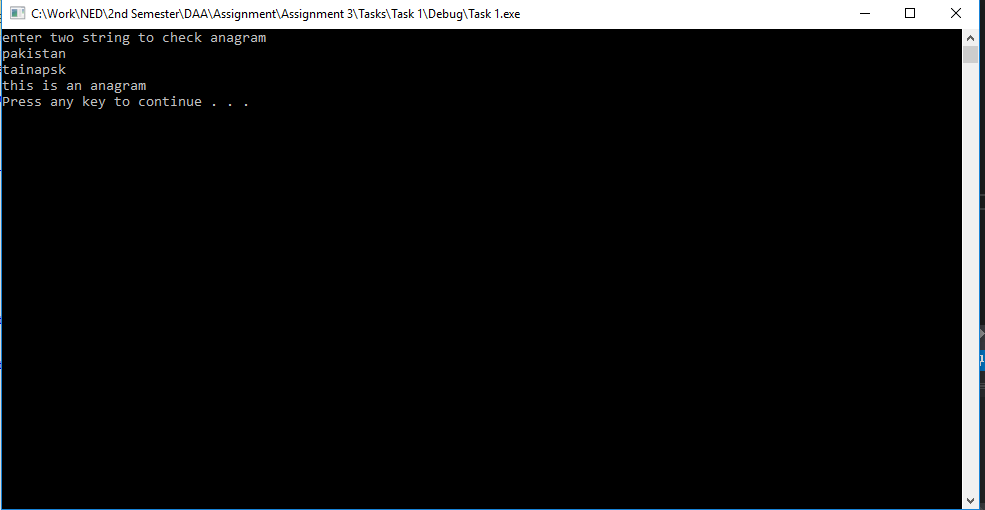
else

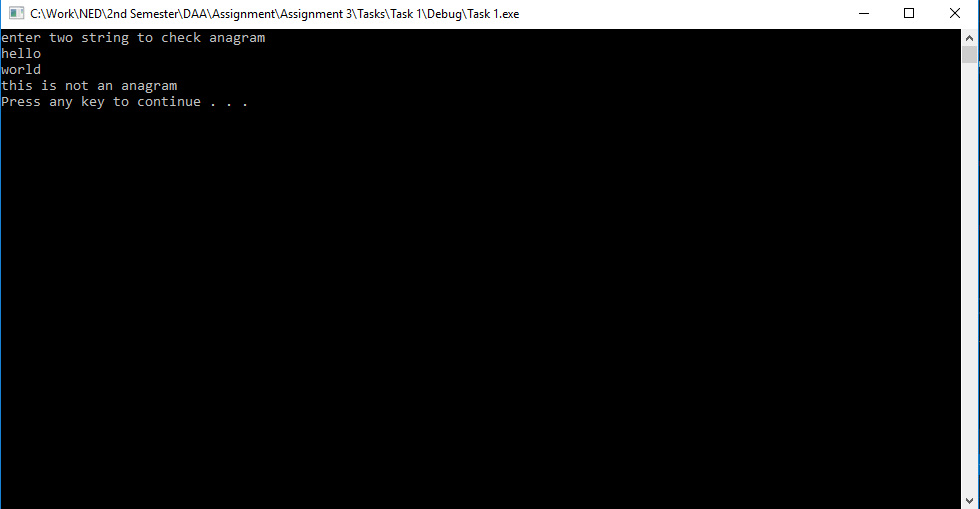
return false;

}

Hence O(n2)

Outputs are given below





**Task 2 (sort and compare)**

Pseudocode is for program

* READ, two strings s1, s2
* LET, size1= length of s1 and size2= length of s2 and I = 0 and j = 0
* IF, size1 = size2 THEN
  + WHILE, I <= size1
    - LET, j = I + 1
    - WHILE, j <= size2
      * IF, s1[i] < s1[j] THEN
        + LET, sw = s1[j] AND s1[j] = s1[i] AND s1[i] = sw
        + END IF
      * IF, s2[i] < s2[j] THEN
        + LET, sw = s2[j] AND s2[j] = s2[i] AND s2[i] = sw
        + END IF
      * END WHILE
    - END WHILE
  + IF, s1 = s2
    - LET, check = true
    - END IF
  + ELSE
    - LET, check = false
  + END IF
* ELSE
  + LET, check = false
* IF, check = true
  + PRINT, this is an anagram
  + END IF
* ELSE
  + PRINT, this is not an anagram

Code in c++

#include<iostream>

#include<string>

using namespace std;

bool anagram(char\*, char\*);

int main()

{

char str1[100], str2[100];

cout << "enter two string to check anagram" << endl;

cin >> str1 >> str2;

bool check = anagram(str1, str2);

if (check == true)

cout << "this is an anagram" << endl;

else

cout << "this is not an anagram" << endl;

system("pause");

}

bool anagram(char\*str1, char\*str2)

{

char sw;

int size1 = strlen(str1), size2 = strlen(str2), i, j;

if (size1 == size2)

{

for (i = 0; i <= size1; i++)

{

for (j = i + 1; j <= size1; j++)

{

if (str1[i] < str1[j])

{

sw = str1[j];

str1[j] = str1[i];

str1[i] = sw;

}

if (str2[i] < str2[j])

{

sw = str2[j];

str2[j] = str2[i];

str2[i] = sw;

}

}

}

if (strcmp(str1, str2) == 0)

return true;

else

return false;

}

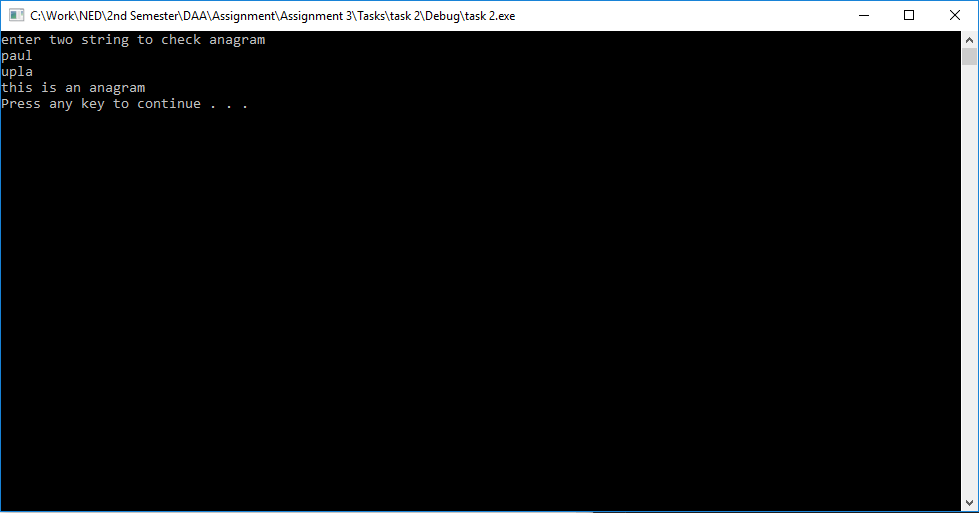
else

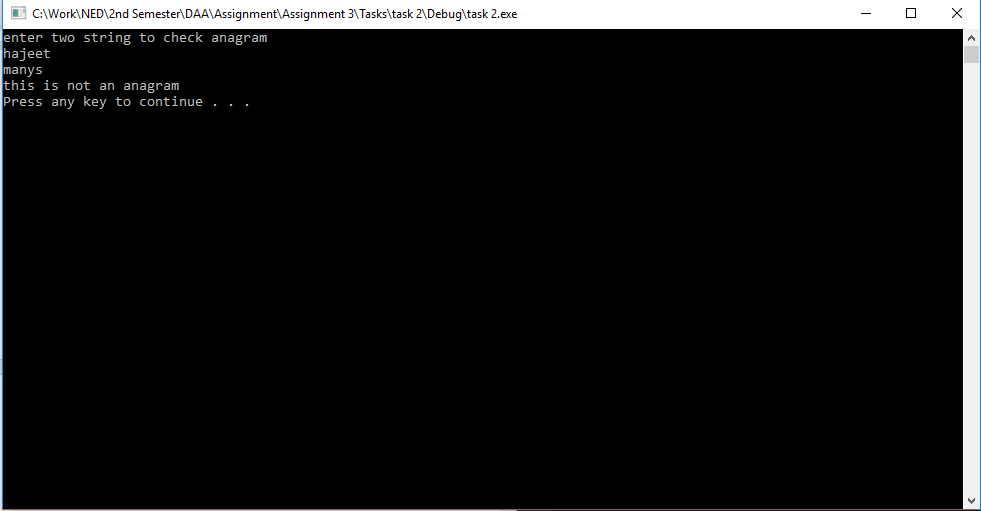
return false;

}

Hence O(n2)

Output is





**Task 3 (count and compare)**

Pseudocode is given below

* READ, two strings s1, s2
* LET, size1 = length of s1 and size2 = length of s2
* IF, size1 = size2 THEN
  + LET, I = 0
  + WHILE, I < size1
    - IF, s1[i] = a THEN
      * LET, alph1[0]++
      * END IF
    - IF, s1[i] = b
      * LET, alph1[1]++
      * END IF
    - IF, s1[i] = c THEN
      * LET, alph1[2]++
      * END IF
    - IF, s1[i] = d THEN
      * LET, alph1[3]++
      * END IF
    - IF, s1[i] = e THEN
      * LET, alph1[4]++
      * END IF
    - IF, s1[i] = f THEN
      * LET, alph1[5]++
      * END IF
    - IF, s1[i] = g THEN
      * LET, alph1[6]++
      * END IF
    - IF, s1[i] = h THEN
      * LET, alph1[7]++
      * END IF
    - IF, s1[i] = i THEN
      * LET, alph1[8]++
      * END IF
    - IF, s1[i] = j THEN
      * LET, alph1[9]++
      * END IF
    - IF, s1[i] = k THEN
      * LET, alph1[10]++
      * END IF
    - IF, s1[i] = l THEN
      * LET, alph1[11]++
      * END IF
    - IF, s1[i] = m THEN
      * LET, alph1[12]++
      * END IF
    - IF, s1[i] = n THEN
      * LET, alph1[13]++
      * END IF
    - IF, s1[i] = o THEN
      * LET, alph1[14]++
      * END IF
    - IF, s1[i] = p THEN
      * LET, alph1[15]++
      * END IF
    - IF, s1[i] = q THEN
      * LET, alph1[16]++
      * END IF
    - IF, s1[i] = r THEN
      * LET, alph1[17]++
      * END IF
    - IF, s1[i] = s THEN
      * LET, alph1[18]++
      * END IF
    - IF, s1[i] = t THEN
      * LET, alph1[19]++
      * END IF
    - IF, s1[i] = u THEN
      * LET, alph1[20]++
      * END IF
    - IF, s1[i] = v THEN
      * LET, alph1[21]++
      * END IF
    - IF, s1[i] = w THEN
      * LET, alph1[22]++
      * END IF
    - IF, s1[i] = x THEN
      * LET, alph1[23]++
      * END IF
    - IF, s1[i] = y THEN
      * LET, alph1[24]++
      * END IF
    - IF, s1[i] = z THEN
      * LET, alph1[25]++
      * END IF
    - IF, s2[i] = a THEN
      * LET, alph2[0]++
      * END IF
    - IF, s2[i] = b
      * LET, alph2[1]++
      * END IF
    - IF, s2[i] = c THEN
      * LET, alph2[2]++
      * END IF
    - IF, s2[i] = d THEN
      * LET, alph2[3]++
      * END IF
    - IF, s2[i] = e THEN
      * LET, alph2[4]++
      * END IF
    - IF, s2[i] = f THEN
      * LET, alph2[5]++
      * END IF
    - IF, s2[i] = g THEN
      * LET, alph2[6]++
      * END IF
    - IF, s2[i] = h THEN
      * LET, alph2[7]++
      * END IF
    - IF, s2[i] = i THEN
      * LET, alph2[8]++
      * END IF
    - IF, s2[i] = j THEN
      * LET, alph2[9]++
      * END IF
    - IF, s2[i] = k THEN
      * LET, alph2[10]++
      * END IF
    - IF, s2[i] = l THEN
      * LET, alph2[11]++
      * END IF
    - IF, s2[i] = m THEN
      * LET, alph2[12]++
      * END IF
    - IF, s2[i] = n THEN
      * LET, alph2[13]++
      * END IF
    - IF, s2[i] = o THEN
      * LET, alph2[14]++
      * END IF
    - IF, s2[i] = p THEN
      * LET, alph2[15]++
      * END IF
    - IF, s2[i] = q THEN
      * LET, alph2[16]++
      * END IF
    - IF, s2[i] = r THEN
      * LET, alph2[17]++
      * END IF
    - IF, s2[i] = s THEN
      * LET, alph2[18]++
      * END IF
    - IF, s2[i] = t THEN
      * LET, alph2[19]++
      * END IF
    - IF, s2[i] = u THEN
      * LET, alph2[20]++
      * END IF
    - IF, s2[i] = v THEN
      * LET, alph2[21]++
      * END IF
    - IF, s2[i] = w THEN
      * LET, alph2[22]++
      * END IF
    - IF, s2[i] = x THEN
      * LET, alph2[23]++
      * END IF
    - IF, s2[i] = y THEN
      * LET, alph2[24]++
      * END IF
    - IF, s2[i] = z THEN
      * LET, alph2[25]++
      * END IF
    - LET, I++
    - END WHILE
  + LET, I = 0
  + WHILE, I < 26
    - IF, alph1[i] != alph2[i]
      * LET, check = false
      * END WHILE
      * END IF
      * END IF
  + LET, check = true
  + END IF
* ELSE
  + LET, check = false

Code in c++

#include<iostream>

#include<string>

using namespace std;

bool anagram(string, string);

int main()

{

string str1, str2;

cout << "enter two string to check anagram" << endl;

cin >> str1 >> str2;

bool check = anagram(str1, str2);

if (check == true)

cout << "this is an anagram" << endl;

else

cout << "this is not an anagram" << endl;

system("pause");

}

bool anagram(string str1, string str2)

{

static int alph1[26], alph2[26];

int size1 = str1.length(), size2 = str2.length(), i;

if (size1 == size2)

{

for (i = 0; i < size1; i++)

{

switch (str1[i])

{

case 'a':

alph1[0]++;

break;

case 'b':

alph1[1]++;

break;

case 'c':

alph1[2]++;

break;

case 'd':

alph1[3]++;

break;

case 'e':

alph1[4]++;

break;

case 'f':

alph1[5]++;

break;

case 'g':

alph1[6]++;

break;

case 'h':

alph1[7]++;

break;

case 'i':

alph1[8]++;

break;

case 'j':

alph1[9]++;

break;

case 'k':

alph1[10]++;

break;

case 'l':

alph1[11]++;

break;

case 'm':

alph1[12]++;

break;

case 'n':

alph1[13]++;

break;

case 'o':

alph1[14]++;

break;

case 'p':

alph1[15]++;

break;

case 'q':

alph1[16]++;

break;

case 'r':

alph1[17]++;

break;

case 's':

alph1[18]++;

break;

case 't':

alph1[19]++;

break;

case 'u':

alph1[20]++;

break;

case 'v':

alph1[21]++;

break;

case 'w':

alph1[22]++;

break;

case 'x':

alph1[23]++;

break;

case 'y':

alph1[24]++;

break;

case 'z':

alph1[25]++;

break;

}

switch (str2[i])

{

case 'a':

alph2[0]++;

break;

case 'b':

alph2[1]++;

break;

case 'c':

alph2[2]++;

break;

case 'd':

alph2[3]++;

break;

case 'e':

alph2[4]++;

break;

case 'f':

alph2[5]++;

break;

case 'g':

alph2[6]++;

break;

case 'h':

alph2[7]++;

break;

case 'i':

alph2[8]++;

break;

case 'j':

alph2[9]++;

break;

case 'k':

alph2[10]++;

break;

case 'l':

alph2[11]++;

break;

case 'm':

alph2[12]++;

break;

case 'n':

alph2[13]++;

break;

case 'o':

alph2[14]++;

break;

case 'p':

alph2[15]++;

break;

case 'q':

alph2[16]++;

break;

case 'r':

alph2[17]++;

break;

case 's':

alph2[18]++;

break;

case 't':

alph2[19]++;

break;

case 'u':

alph2[20]++;

break;

case 'v':

alph2[21]++;

break;

case 'w':

alph2[22]++;

break;

case 'x':

alph2[23]++;

break;

case 'y':

alph2[24]++;

break;

case 'z':

alph2[25]++;

break;

}

}

for (i = 0; i < 26; i++)

{

if (alph1[i] == alph2[i])

continue;

else

return false;

}

return true;

}

else

return false;

}

Hence O(n)

Output

