**CSE2012-LAB**

**NAME:** O G RAGAVI

**REG NO:** 20BCE1988

**TITLE:** STRING MATCHING ALGORITHMS(RABIN-KARP)

**EX NO:** 9

**IN\_LAB PRACTICE SESSION:**

**1.NAIVE STRING MATCHING ALGORITHM:**

**CODE:**

#include<iostream>

using namespace std;

#include<string>

bool check\_pattern(string T, string P, int s)

{

int m = P.length(),i;

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

{

if(P[i]!=T[s+i+1])

return false;

}

return true;

}

void naive\_string\_matcher(string T, string P)

{

int n,m,s,i,count=0;

n = T.length();

m = P.length();

for(s=-1;s<=n-m-1;s++)

{

if (check\_pattern(T,P,s))

{

cout<<"Pattern occurs at "<<s+1<<endl;

count=count+1;

}

}

cout<<"count is "<<count<<endl;

}

int main()

{

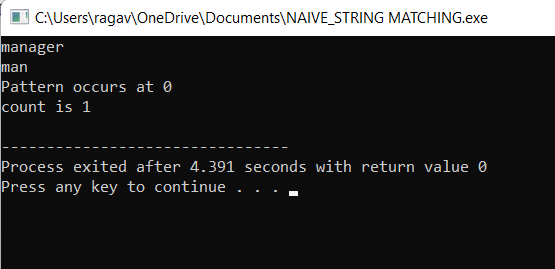
string T, P;

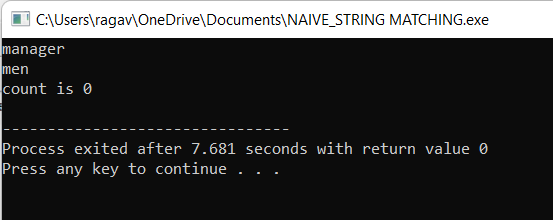
cin>>T>>P;

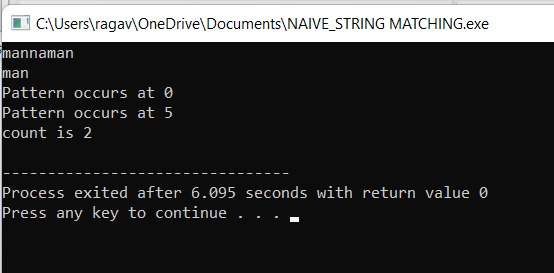
naive\_string\_matcher(T,P);

}

OUTPUT:







**LAB PRACTICE SHEET:**

**1.C++ IMPLEMENTATION OF RABIN KARP:**CODE:

#include<iostream>

using namespace std;

#include<string>

#include<math.h>

int valueOf(char ch)

{

return (ch-'0');

}

bool check\_characters(string T, string P, int s)

{

int n,m,i;

n = T.length();

m = P.length();

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

{

if(P[i]!=T[s+i+1])

return false;

}

return true;

}

void rabin\_karp\_matcher(string T, string P, int d, int q)

{

int n, m, h, p, t0, i,s;

n = T.length();

m = P.length();

h = (int)pow(d,m-1)%q;

cout<<"h value is "<<h<<endl;

p = 0;

t0 = 0;

int count=0;

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

{

p = (d\*p + valueOf(P[i]))%q;

t0 = (d\*t0 + valueOf(T[i]))%q;

}

for(s=-1;s<n-m;s++)

{

cout<<p<<" "<<t0<<endl;

if(p==t0)

{

if (check\_characters(T,P,s))

{

cout<<"Pattern occurs at index "<<s+1<<endl;

count=count+1;

}

}

if(s<n-m)

t0 = (d\*(t0-valueOf(T[s+1])\*h)+valueOf(T[s+m+1]))%q;

}

if(count==0)

printf("pattern does not occur!!!");

else

printf("pattern occurs %d times!!!",count);

}

int main()

{

string T, P;

int d,q;

cin>>T>>P;

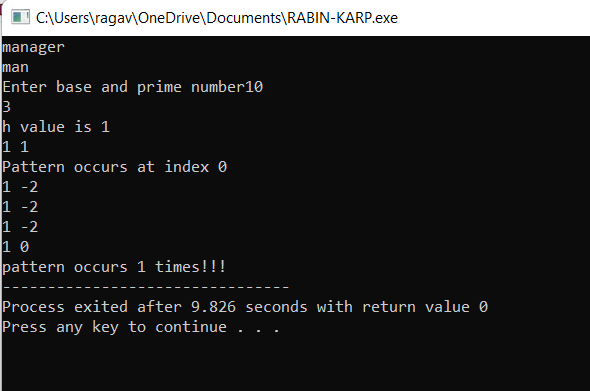
cout<<"Enter base and prime number";

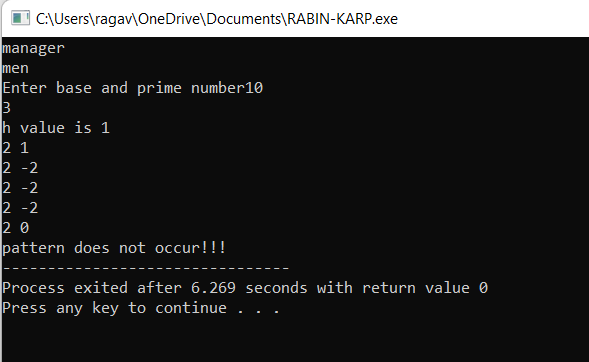
cin>>d>>q;

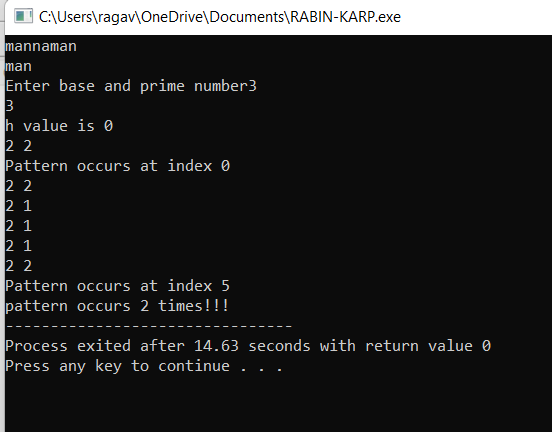
rabin\_karp\_matcher(T,P,d,q);

}

OUTPUT:

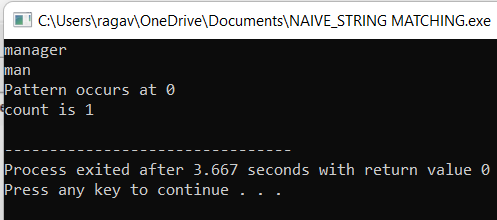


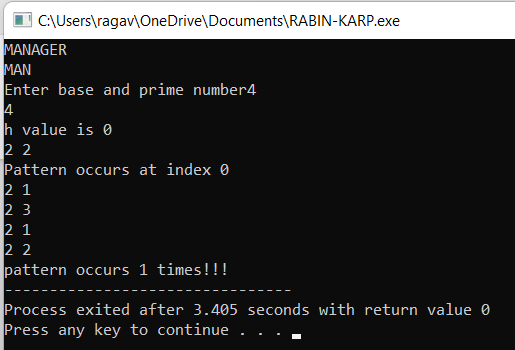


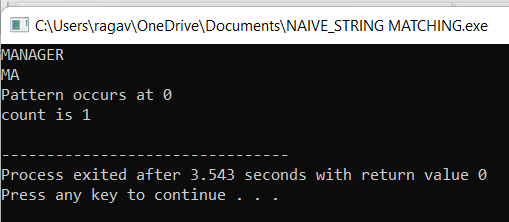


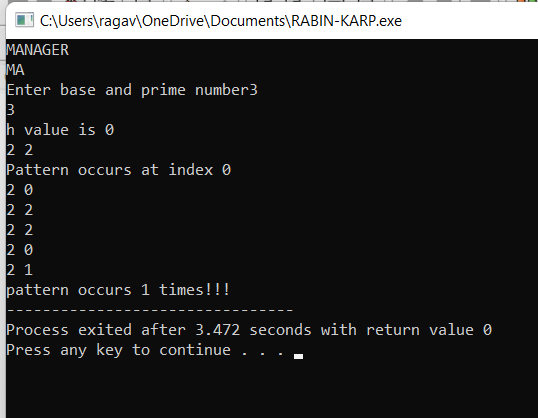
2**.RUNNING TIME COMPUTATION**:

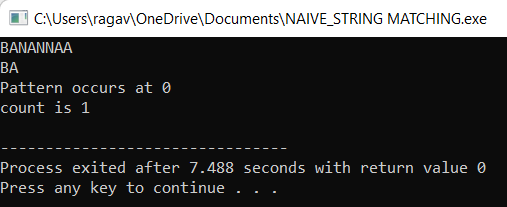
|  |  |  |  |
| --- | --- | --- | --- |
| m | n | T1(P) | T2(p) |
| 7 | 3 | 3.667 | 3.405 |
| 7 | 2 | 3.543 | 3.472 |
| 8 | 4 | 7.488 | 5.842 |
| 8 | 3 | 6.317 | 7.033 |
| 10 | 2 | 8.753 | 9.861 |

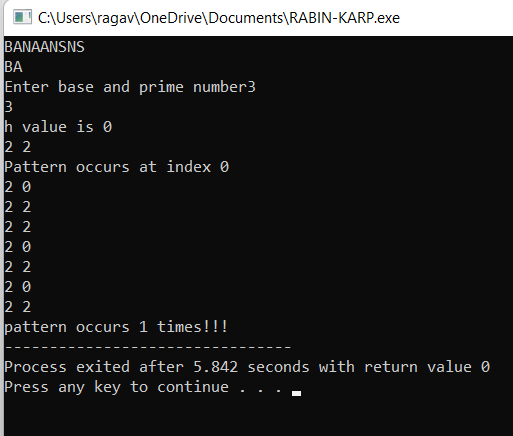


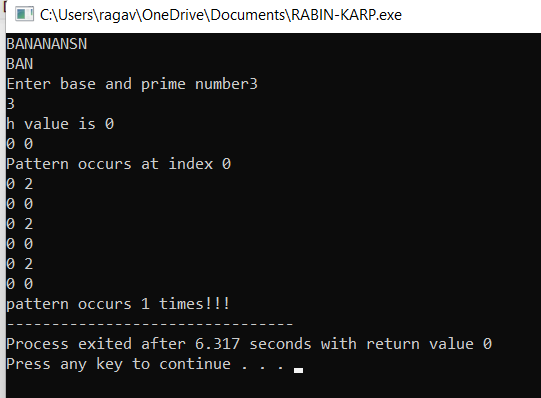


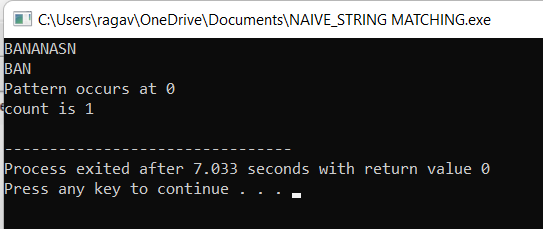


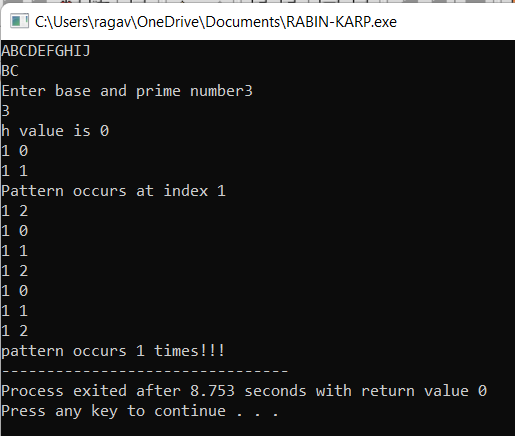


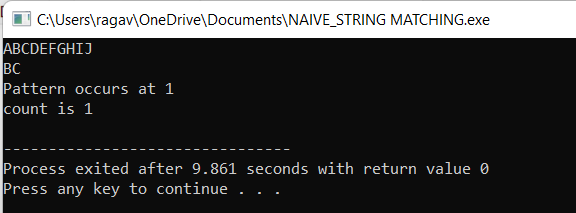




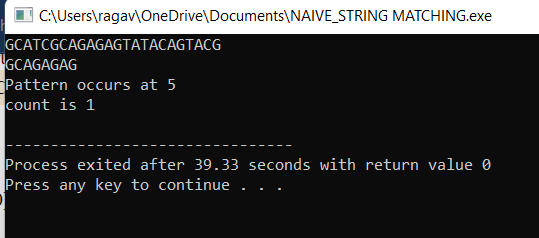






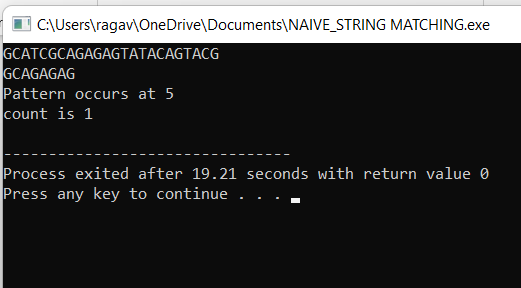


3.Q)



4.Q)

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |  |  |



**5.Q) PRINT THE NO OF COMPARISONS MADE(IPS):**

**CODE:**

#include<iostream>

using namespace std;

#include<string>

bool check\_pattern(string T, string P, int s)

{

int m = P.length(),i;

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

{

if(P[i]!=T[s+i+1])

return false;

}

return true;

}

void naive\_string\_matcher(string T, string P)

{

int n,m,s,i,count=0;

n = T.length();

m = P.length();

for(s=-1;s<=n-m-1;s++)

{

count=count+1;

if (check\_pattern(T,P,s))

{

cout<<"Pattern occurs at "<<s+1<<endl;

//count=count+1;

}

}

cout<<"count of the no of comaparisons is :"<<count<<endl;

}

int main()

{

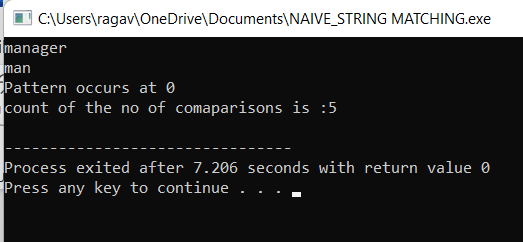
string T, P;

cin>>T>>P;

naive\_string\_matcher(T,P);

}

**OUTPUT:**



**6.Q) PRINT THE NO OF COMPARISONS MADE(LPS):**

**CODE:**

#include<iostream>

using namespace std;

#include<string>

#include<math.h>

int valueOf(char ch)

{

return (ch-'0');

}

bool check\_characters(string T, string P, int s)

{

int n,m,i;

n = T.length();

m = P.length();

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

{

if(P[i]!=T[s+i+1])

return false;

}

return true;

}

void rabin\_karp\_matcher(string T, string P, int d, int q)

{

int n, m, h, p, t0, i,s;

n = T.length();

m = P.length();

h = (int)pow(d,m-1)%q;

cout<<"h value is "<<h<<endl;

p = 0;

t0 = 0;

int count=0;

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

{

p = (d\*p + valueOf(P[i]))%q;

t0 = (d\*t0 + valueOf(T[i]))%q;

}

for(s=-1;s<n-m;s++)

{

count=count+1;

cout<<p<<" "<<t0<<endl;

if(p==t0)

{

if (check\_characters(T,P,s))

{

cout<<"Pattern occurs at index "<<s+1<<endl;

//count=count+1;

}

}

if(s<n-m)

t0 = (d\*(t0-valueOf(T[s+1])\*h)+valueOf(T[s+m+1]))%q;

}

/\*if(count==0)

printf("pattern does not occur!!!");

else

printf("pattern occurs %d times!!!",count);\*/

cout<<"the no of comparisons made is : "<<count<<endl;

}

int main()

{

string T, P;

int d,q;

cin>>T>>P;

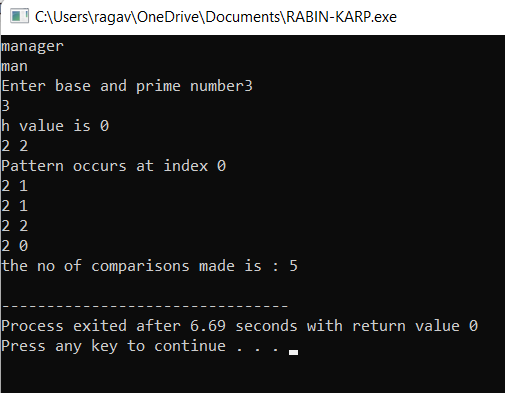
cout<<"Enter base and prime number";

cin>>d>>q;

rabin\_karp\_matcher(T,P,d,q);

}

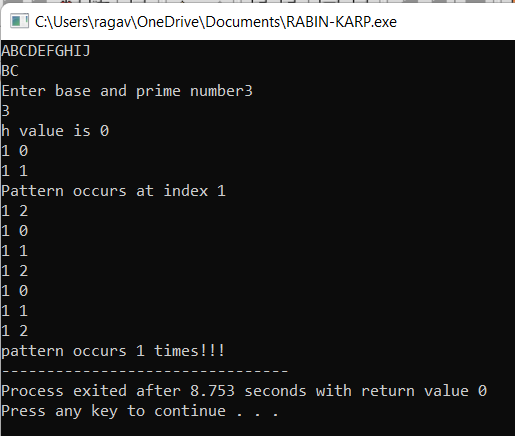
**OUTPUT:**



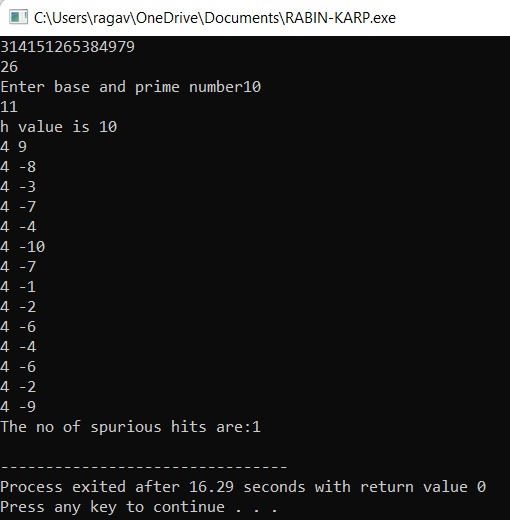
7**.comparison of brute force and Rabin-karp:**

|  |  |  |  |
| --- | --- | --- | --- |
| m | n | C1(p) | C2(p) |
| 7 | 3 | 5 | 5 |
| 9 | 2 | 8 | 8 |
| 10 | 2 | 1 | 1 |
| 11 | 3 | 2 | 2 |

8.Q) **Worst case-if the text length is large but the pattern length is small.**



9.Q)

. 

10.Q)

