**Cardinality of a Set**

**Code**

#include<bits/stdc++.h>

using namespace std;

int CheckDuplicate(char\* c,int l)

{

int co=0,x=l,m;

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

{

for(int j=i+1; j<l; j++)

{

if(c[i] == c[j])

{

co++;

break;

}

}

if(i==(l-1)){

c[i]='\0';

break;

}

}

return co;

}

int main()

{

int p,len,newlen,w;

char c[40],d[40];

cout<<"Enter the set: ";

gets(c);

len = strlen(c);

newlen = ((len-2)/2)+1;

for(int i=0,j=0;j!=newlen;i++)

{

if(('0'<=c[i] && c[i]<='9') || ('a'<=c[i] && c[i]<='z') || ('A'<=c[i] && c[i]<='Z')){

d[j]=c[i];

j++;

}

if(j==(newlen)) {

d[j]='\0';

break;

}

}

w = strlen(d);

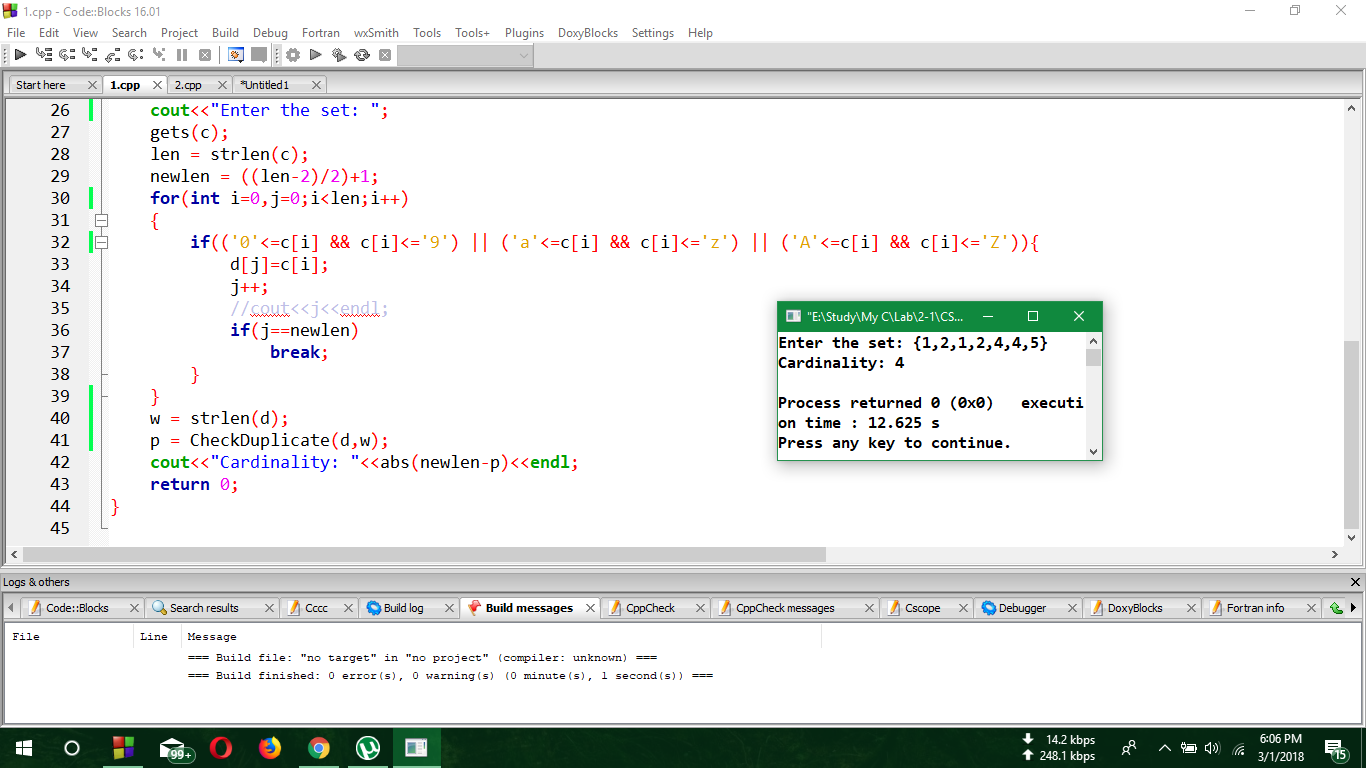
p = CheckDuplicate(d,w);

cout<<"Cardinality: "<<abs(newlen-p)<<endl;

return 0;

}

**Output**



**Equivalency of two sets**

**Code**

#include<bits/stdc++.h>

using namespace std;

char \*GetDistinct(char \*str)

{

int resIndex=1, ipIndex=1;

while (\*(str+ipIndex))

{

if (\*(str+ipIndex)!=\*(str+ipIndex-1))

{

\*(str+resIndex)=\*(str+ipIndex);

resIndex++;

}

ipIndex++;

}

\*(str+resIndex)='\0';

return str;

}

int main()

{

int ActLen1,ActLen2,c=0,d=0;

char set1[30],set2[30];

char str1[15],str2[15];

char \*s1,\*s2;

while(1)

{

cout<<"Enter Set A: ";

gets(set1);

cout<<"Enter Set B: ";

gets(set2);

ActLen1=((strlen(set1)-2)/2)+1;

ActLen2=((strlen(set2)-2)/2)+1;

for(int i=0,j=0;i<strlen(set1);i++)

{

if(j==ActLen1){

str1[j]='\0';

break;

}

else if(set1[i]!='{' && set1[i]!='}' && set1[i]!=','){

str1[j]=set1[i];

j++;

}

}

for(int i=0,j=0;i<strlen(set2);i++)

{

if(j==ActLen2){

str2[j]='\0';

break;

}

else if(set2[i]!='{' && set2[i]!='}' && set2[i]!=','){

str2[j]=set2[i];

j++;

}

}

sort(str1,str1+ActLen1);

sort(str2,str2+ActLen2);

s1=GetDistinct(str1);

s2=GetDistinct(str2);

if(strlen(s1)==strlen(s2)){

for(int i=0;i<strlen(s1);i++)

{

for(int j=0;j<strlen(s1);j++)

{

if(s1[i]==s2[j])

c++;

}

}

if(c==strlen(s1))

cout<<"Set A and B are equivalent\n"<<endl;

else

cout<<"Set A and B are not equivalent\n"<<endl;

}

else

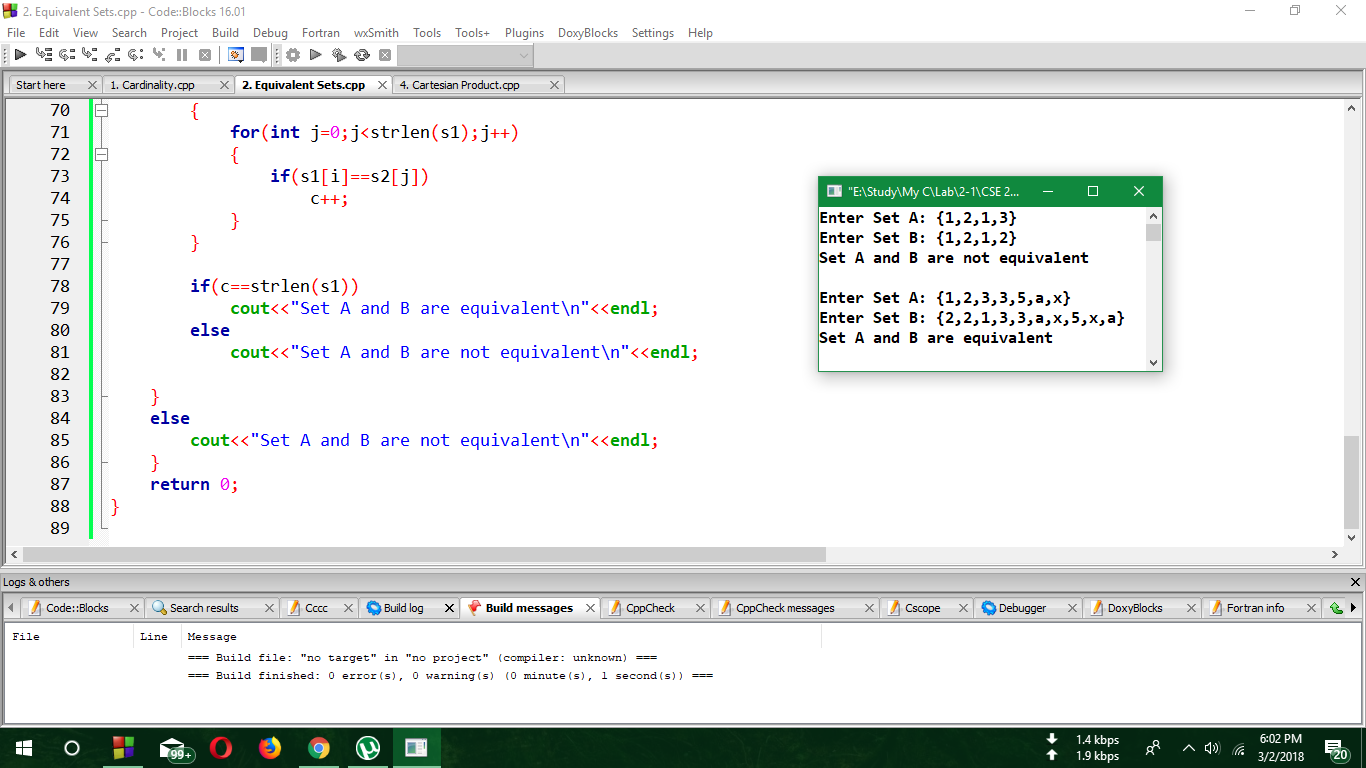
cout<<"Set A and B are not equivalent\n"<<endl;

}

return 0;

}

**Output**



**Power set of a set**

**Code**

#include<bits/stdc++.h>

using namespace std;

#define PowSetSize(SetSize) pow(2,SetSize)

int main()

{

char set1[30],str1[15];

cout<<"Enter a Set: ";

gets(set1);

int ActLen1=((strlen(set1)-2)/2)+1;

for(int i=0,j=0;i<strlen(set1);i++)

{

if(j==ActLen1){

str1[j]='\0';

break;

}

else if(set1[i]!='{' && set1[i]!='}' && set1[i]!=','){

str1[j]=set1[i];

j++;

}

}

int SetSize = strlen(str1);

cout<<"The Power Set of input set is: ";

cout<<"{ {";

for(int i=0;i<PowSetSize(SetSize);i++)

{

for(int j=0;j<SetSize;j++)

{

if((i&1<<j)==0){

cout<<" "<<str1[j] <<" ";

}

}

if(i!=(PowSetSize(SetSize)-1))

cout<<"}"<<','<<"{";

else

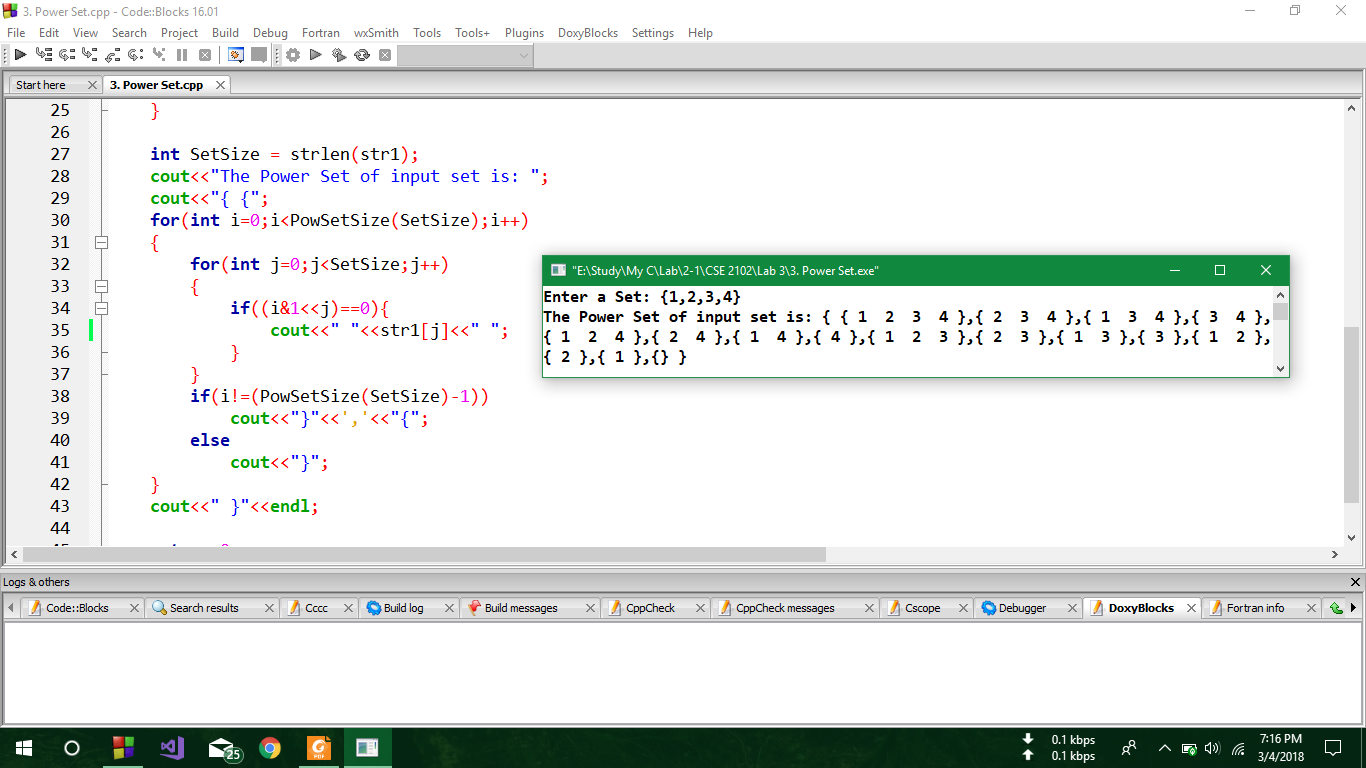
cout<<"}";

}

cout<<" }"<<endl;

return 0;

}

**Output**

**Cartesian product of two sets**

**Code**

#include<bits/stdc++.h>

using namespace std;

int main()

{

char set1[30],set2[30],str1[15],str2[15],CP[100];

int len1,len2,ActLen1,ActLen2,k=0;

cout<<"Enter set A: ";

cin>>set1;

cout<<"Enter set B: ";

cin>>set2;

len1=strlen(set1);

len2=strlen(set2);

ActLen1=((len1-2)/2)+1;

ActLen2=((len2-2)/2)+1;

for(int i=0,j=0;i<len1;i++)

{

if(j==ActLen1)

break;

else if(set1[i]!='{' && set1[i]!='}' && set1[i]!=','){

str1[j]=set1[i];

j++;

}

}

for(int i=0,j=0;i<len2;i++)

{

if(j==ActLen2)

break;

else if(set2[i]!='{' && set2[i]!='}' && set2[i]!=','){

str2[j]=set2[i];

j++;

}

}

for(int i=0;i<ActLen1;i++)

{

for(int j=0;j<ActLen2;j++)

{

CP[k]='(';

CP[k+1]=str1[i];

CP[k+2]=',';

CP[k+3]=str2[j];

CP[k+4]=')';

if(j!=(ActLen2-1)||i!=(ActLen1-1)){

CP[k+5]=',';

k=k+6;

}

else

k=k+5;

}

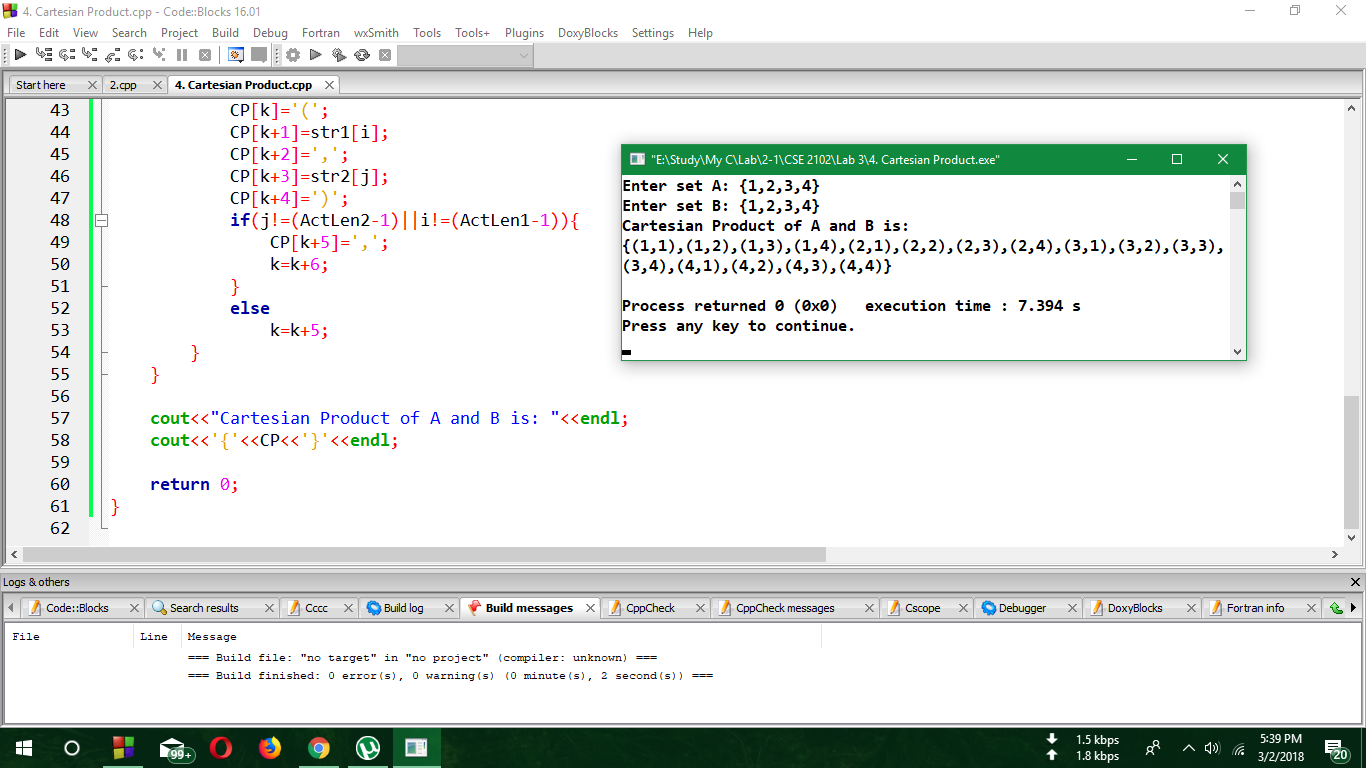
}

cout<<"Cartesian Product of A and B is: "<<endl;

cout<<'{'<<CP<<'}'<<endl;

return 0;

}

**Output**