BRACT's
VISHWAKARMA INSTITUTE OF INFORMATION TECHNOLOGY, PUNE – 48
An Autonomous Institute Affiliated to Savitribai Phule Pune University, Pune

SD(LP-II) ASSIGNMENT (S.Y.B. Tech. – DIV: C)

Name: Prajwal Dabhade;

Roll no.: 223012;

GR. No.: 17u385; Batch: C1

*****************************Assignment No.1*********************************

- **Aim**: To create ADT that implement the set concept.

a) add (newelement) –place a value into the set.
b) remove (element).
c) contains (element) - return true if element is in collection.
d) size() - return number of values in collection.
e) intersection of two sets.
f) difference between two sets.
g) subset.

- **Objective:** We have to implement this using basic data structure.
- **Theory:** In computer science, a set is an abstract data type that can store unique values, without any particular order. It is a computer implementation of the mathematical concept of a finite set. some set data structures are designed for static or frozen sets that do not change after they are constructed.
  Applications: Hash function & Spelling checker.
- **Algorithm**:

1. Intersection is an associative operation; that is, for any sets A, B, and C, one has A n (B n C) = (A n B) n C. Intersection is also commutative; for any A and B, one has A n B = B n A. It thus makes sense to talk about intersections of multiple sets. The intersection of A, B, C, and D, for example, is unambiguously written A n B n C n D.
2. We say that A and B are disjoint if A does not intersect B. In plain language, they have no elements in common. A and B are disjoint if their intersection is empty, denoted
3. The union of two sets A and B is the set of elements which are in A, in B, or in both A and B. In symbols,

- **Program:**

```
#include<iostream>

using namespace std;

struct set

{

int s[10],size;

}s1,s2,s3,s4,s5;


void insert()

{int ch,key;

cout<<"\n Enter element to be insert : ";cin>>key;

cout<<"\n Which set : ";cin>>ch;

switch(ch)

{

case 1:s1.s[s1.size++]=key;break;

case 2:s2.s[s2.size++]=key;break;

default:cout<<"\nWrong set ";

}}

void remove()

{

int ch,key,i;

cout<<"\n Enter element to be remove : ";cin>>key;

cout<<"\n Which set : ";cin>>ch;

switch(ch)

{
```

```
case 1:for(i=0;i<10 && s1.s[i]!=key;i++)
        {}
        while(i<10)
        {s1.s[i]=s1.s[i+1];i++;}
        s1.size--;
        break;
case 2:for(i=0;i<10 && s1.s[i]!=key;i++){}
        while(i<10)
        {s2.s[i]=s2.s[i+1];i++;}
        s2.size--;
        break;
default:cout<<"\nWrong set ";
}}
void contain()
{
int ch,key,i,f=0;
cout<<"\n Enter element to be search : ";cin>>key;
cout<<"\n Which set : ";cin>>ch;
switch(ch)
{
case 1:for(i=0;i<10;i++)
        {
         if(s1.s[i]==key)
         {cout<<endl<<key<<" is present in set 1.";f=1;break;}
         }
        if(f==0)
        cout<<endl<<key<<" is not present in set 1.";
        break;
case 2:for(i=0;i<10;i++)
        {
         if(s2.s[i]==key)
         {cout<<endl<<key<<" is present in set 2.";f=1;break;}
         }
        if(f==0)
        cout<<endl<<key<<" is not present in set 2.";
        break;
default:cout<<"\nWrong set ";
}}
void size()
{
cout<<"\n size of set 1 : "<<s1.size;
cout<<"\n size of set 2 : "<<s2.size;
cout<<endl;
}
void printset()
{int i;
cout<<endl<<"set1 :";;
for(i=0;i<s1.size;i++)
cout<<" "<<s1.s[i];
cout<<endl<<"set2 :";;
```

```
for(i=0;i<s2.size;i++)
cout<<" "<<s2.s[i];
cout<<endl;
}
void intersect()
{
int i,j,f=0;
s3.size=0;
for(i=0;i<s1.size;i++)
{
for(j=0;j<s2.size;j++)
{
if(s1.s[i]==s2.s[j])
{s3.s[s3.size++]=s1.s[i];f=1;}
}}
cout<<"\n intersection of set : ";
printset();
cout<<"\n is ";
if(f==0)
cout<<"NULL. \n";
else
{
for(i=0;i<s3.size;i++)
cout<<" "<<s3.s[i];
cout<<endl;
}}
void uni()
{
int i,j,f;
s4.size=0;
for(i=0;i<s1.size;i++)
{s4.s[s4.size++]=s1.s[i];}

for(i=0;i<s2.size;i++)
{f=0;
for(j=0;j<s1.size;j++)
{
if(s2.s[i]==s1.s[j])
{f=1;}
}
if(f==0)
{s4.s[s4.size++]=s2.s[i];}
}
cout<<"\n union of set : ";
printset();
cout<<"\n is ";
for(i=0;i<s4.size;i++)
cout<<" "<<s4.s[i];
cout<<endl;
}
```

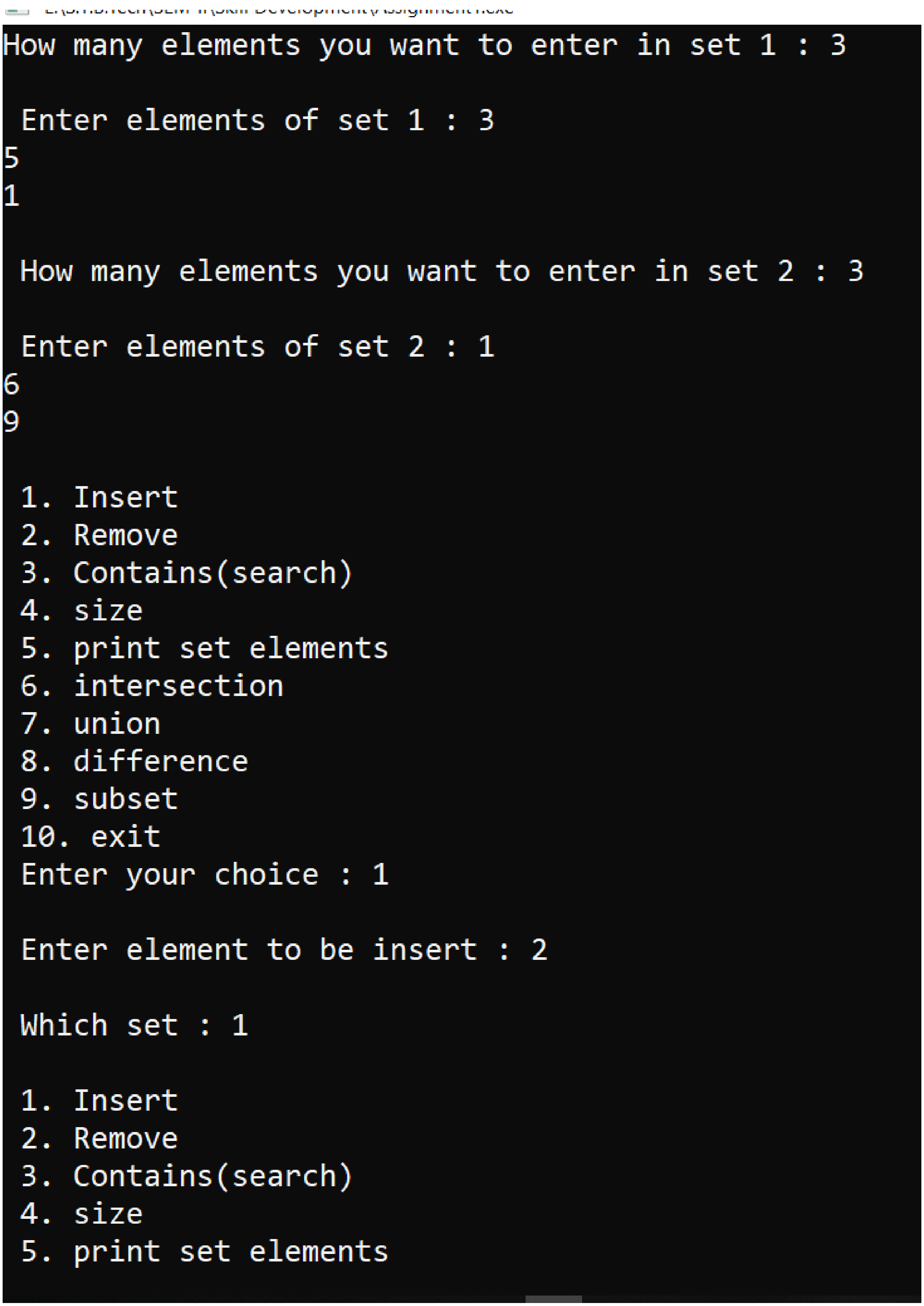
```
void difference()
{
int i,j,f;
cout<<"\n set1-set2 is ";
for(i=0;i<s1.size;i++)
{f=0;
for(j=0;j<s2.size;j++)
{
if(s1.s[i]==s2.s[j])
f=1;
}
if(f==0)
cout<<s1.s[i]<<" ";
}
cout<<endl;
cout<<"\n set2-set1 is ";
for(i=0;i<s2.size;i++)
{f=0;
for(j=0;j<s1.size;j++)
{
if(s2.s[i]==s1.s[j])
f=1;
}
if(f==0)
cout<<s2.s[i]<<" ";
```

```
}}
void subset()
{
int i,j,f,m;
for(i=0;i<s1.size;i++)
{f=0;
for(j=0;j<s2.size;j++)
{
if(s1.s[i]==s2.s[j])
{f=1;}
}
if(f==0)
{cout<<"\nset1 is not subset of set 2.\n";break;}
}
if(f==1)
{cout<<"\nset1 is subset of set 2.\n";}
for(i=0;i<s2.size;i++)
{f=0;
for(j=0;j<s1.size;j++)
{
if(s2.s[i]==s1.s[j])
{f=1;}
}
if(f==0)
{cout<<"\nset2 is not subset of set 1.\n";break;}
```

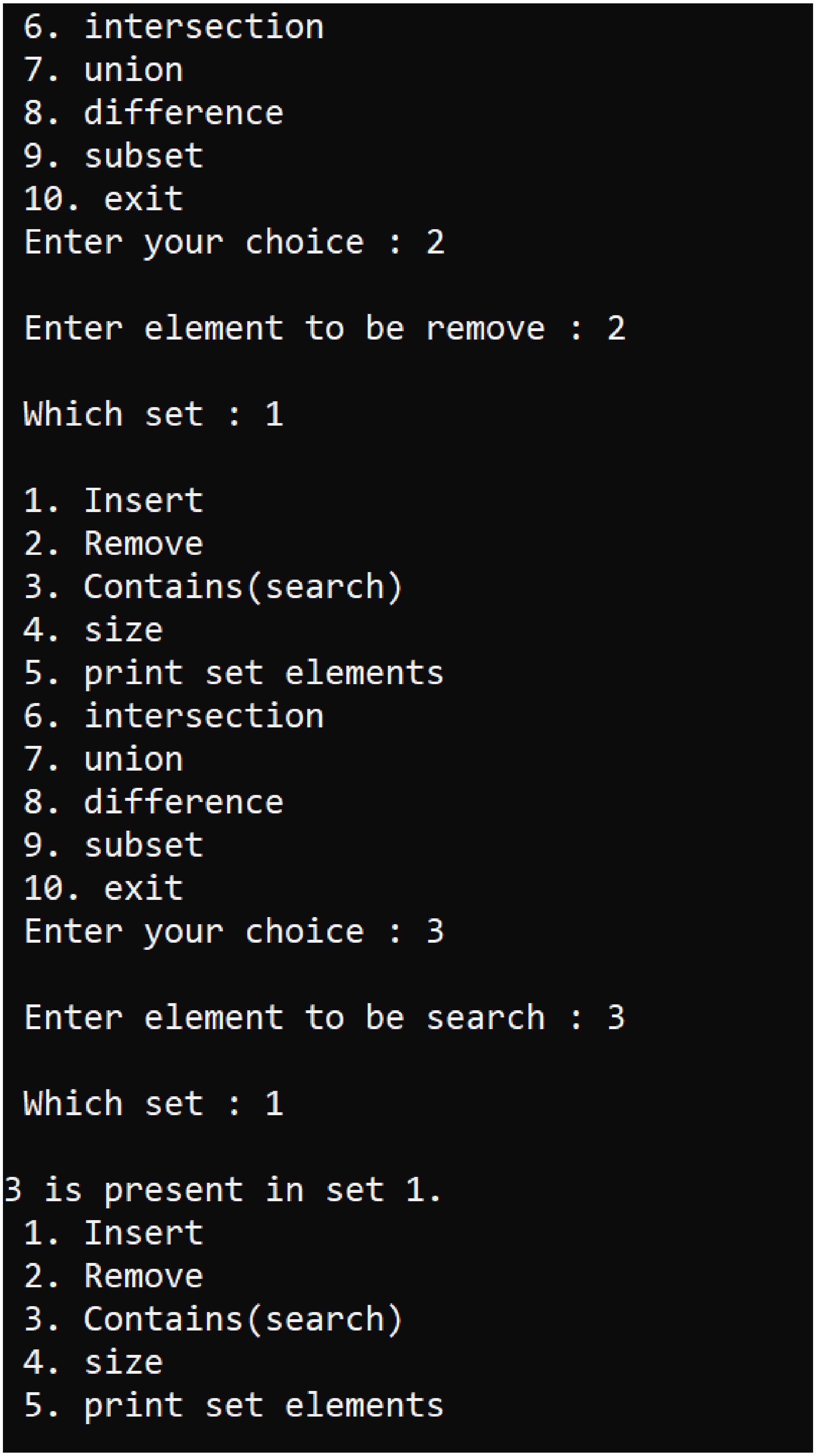
```
}

if(f==1)

{cout<<"\nset2 is subset of set 1.\n";}

}

int main()

{

int i,ch,m,n;

cout<<"How many elements you want to enter in set 1 : ";cin>>m;

s1.size=m;

cout<<" \n Enter elements of set 1 : ";

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

cin>>s1.s[i];

cout<<" \n How many elements you want to enter in set 2 : ";cin>>n;

s2.size=n;

cout<<" \n Enter elements of set 2 : ";

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

cin>>s2.s[i];

while(1)

{

cout<<"\n 1. Insert \n 2. Remove \n 3. Contains(search) \n 4. size \n 5. print set elements \n 6. intersection \n 7. union \n 8. difference \n 9. subset \n 10. exit \n Enter your choice : ";

cin>>ch;

if(ch==10)

break;

switch(ch)
```

```
{
case 1:insert();break;
case 2:remove();break;
case 3:contain();break;
case 4:size();break;
case 5:printset();break;
case 6:intersect();break;
case 7:uni();break;
case 8:difference();break;
case 9:subset();break;
default:cout<<"Wrong choice";
}}
return 0;
}
```

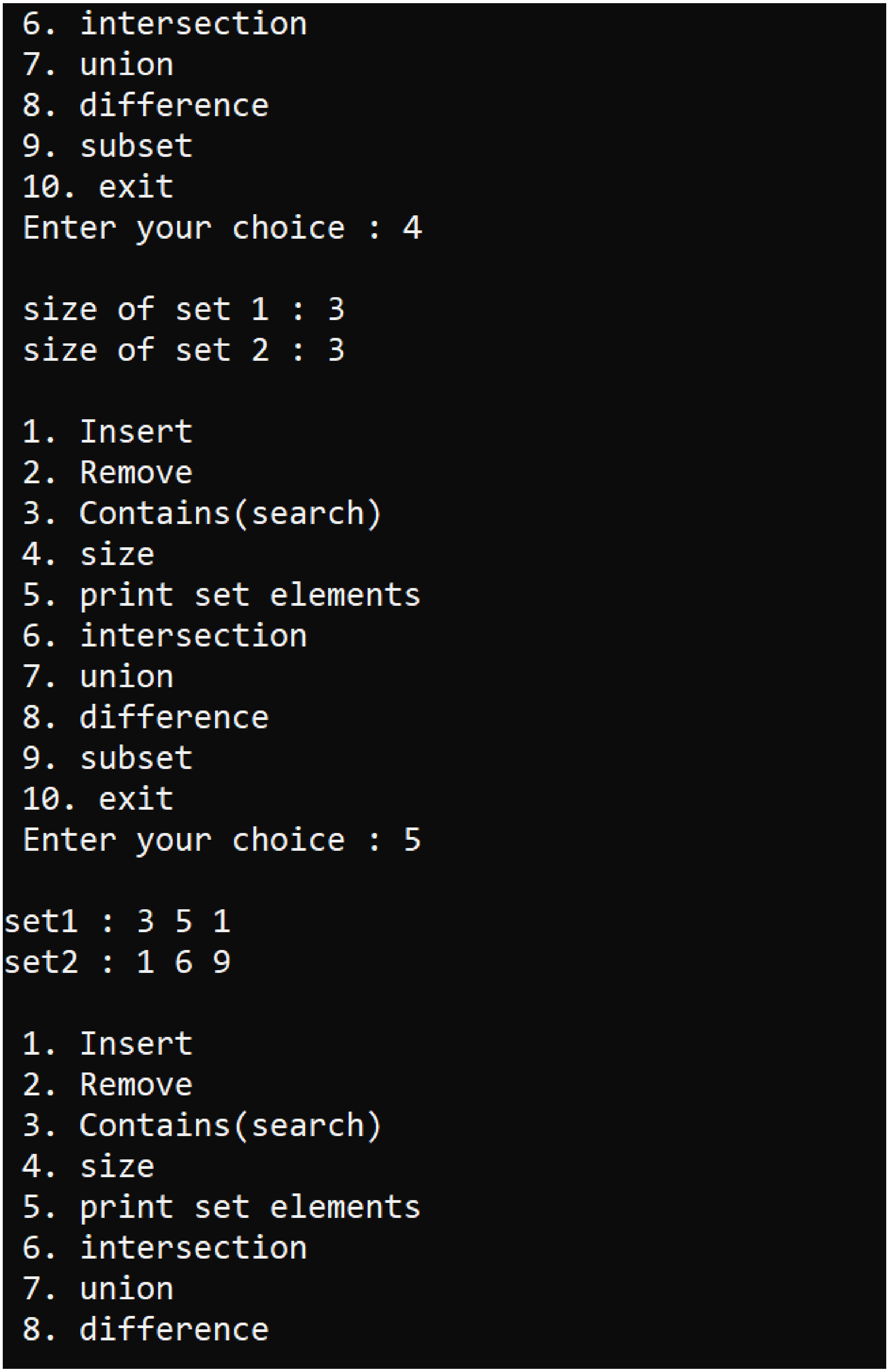
- **Output:**



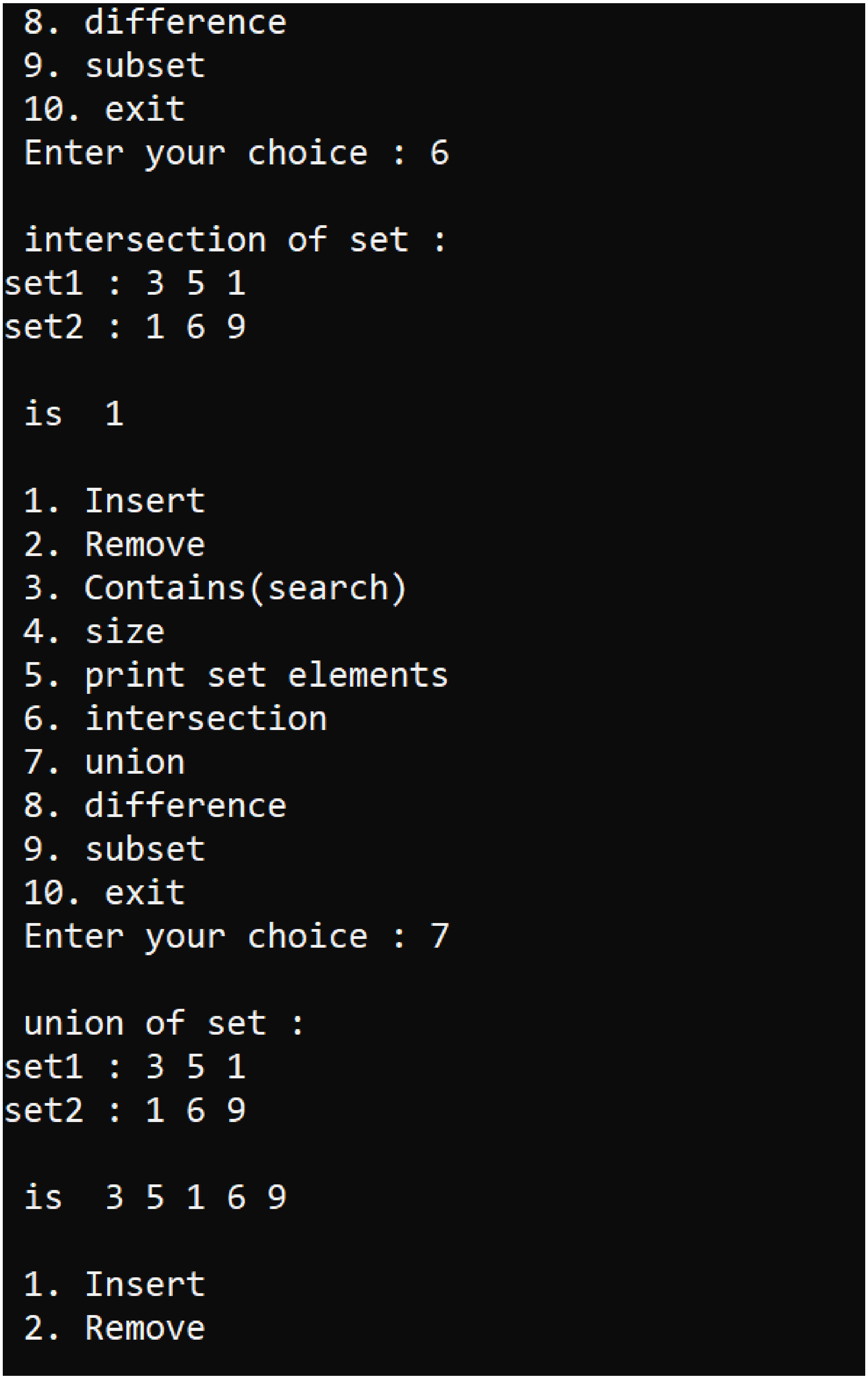
```
How many elements you want to enter in set 1 : 3

 Enter elements of set 1 : 3
5
1

 How many elements you want to enter in set 2 : 3

 Enter elements of set 2 : 1
6
9

 1. Insert
 2. Remove
 3. Contains(search)
 4. size
 5. print set elements
 6. intersection
 7. union
 8. difference
 9. subset
 10. exit
 Enter your choice : 1

 Enter element to be insert : 2

 Which set : 1

 1. Insert
 2. Remove
 3. Contains(search)
 4. size
 5. print set elements
```



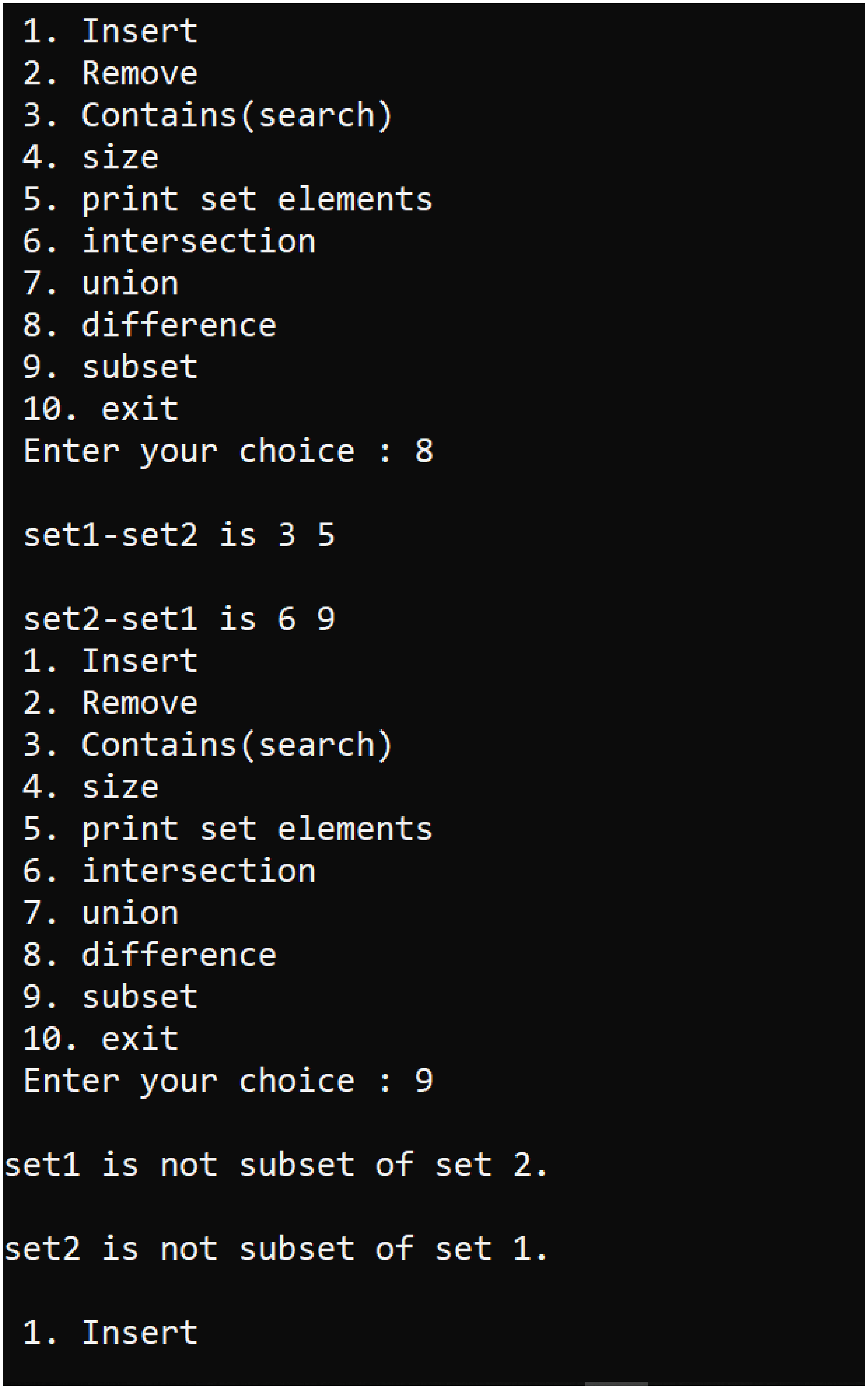
```
 6. intersection
 7. union
 8. difference
 9. subset
 10. exit
 Enter your choice : 2

 Enter element to be remove : 2

 Which set : 1

 1. Insert
 2. Remove
 3. Contains(search)
 4. size
 5. print set elements
 6. intersection
 7. union
 8. difference
 9. subset
 10. exit
 Enter your choice : 3

 Enter element to be search : 3

 Which set : 1

3 is present in set 1.
 1. Insert
 2. Remove
 3. Contains(search)
 4. size
 5. print set elements
```



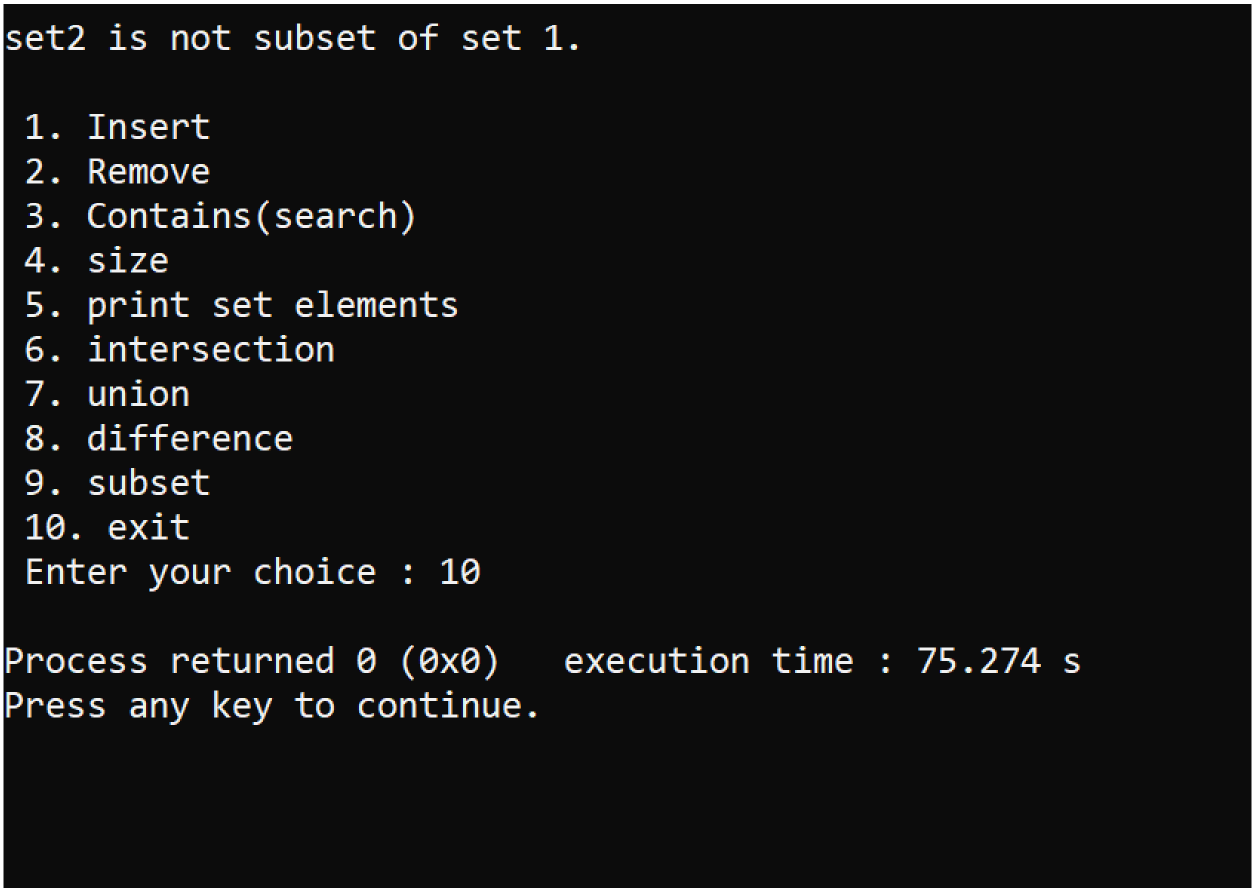
```
 6. intersection
 7. union
 8. difference
 9. subset
 10. exit
 Enter your choice : 4

 size of set 1 : 3
 size of set 2 : 3

 1. Insert
 2. Remove
 3. Contains(search)
 4. size
 5. print set elements
 6. intersection
 7. union
 8. difference
 9. subset
 10. exit
 Enter your choice : 5

set1 : 3 5 1
set2 : 1 6 9

 1. Insert
 2. Remove
 3. Contains(search)
 4. size
 5. print set elements
 6. intersection
 7. union
 8. difference
```



```
 8. difference
 9. subset
 10. exit
 Enter your choice : 6

 intersection of set :
set1 : 3 5 1
set2 : 1 6 9

 is  1

 1. Insert
 2. Remove
 3. Contains(search)
 4. size
 5. print set elements
 6. intersection
 7. union
 8. difference
 9. subset
 10. exit
 Enter your choice : 7

 union of set :
set1 : 3 5 1
set2 : 1 6 9

 is  3 5 1 6 9

 1. Insert
 2. Remove
```



```
 1. Insert
 2. Remove
 3. Contains(search)
 4. size
 5. print set elements
 6. intersection
 7. union
 8. difference
 9. subset
 10. exit
 Enter your choice : 8

 set1-set2 is 3 5

 set2-set1 is 6 9
 1. Insert
 2. Remove
 3. Contains(search)
 4. size
 5. print set elements
 6. intersection
 7. union
 8. difference
 9. subset
 10. exit
 Enter your choice : 9

set1 is not subset of set 2.

set2 is not subset of set 1.

 1. Insert
```



```
set2 is not subset of set 1.

 1. Insert
 2. Remove
 3. Contains(search)
 4. size
 5. print set elements
 6. intersection
 7. union
 8. difference
 9. subset
 10. exit
 Enter your choice : 10

Process returned 0 (0x0)   execution time : 75.274 s
Press any key to continue.
```

- **Conclusion:** Thus, we have studied set theory using basic data structure.