EXPERIMENT 9

- ASHISH KUMAR
- 2K18/SE/041

AIM:- Write a program to perform an experiment on Mutation Testing.

THEORY:- **Mutation Testing** is a type of Software Testing that is performed to design new software tests and also evaluate the quality of already existing software tests. Mutation testing is related to modification a program in small ways. It focuses to help the tester develop effective tests or locate weaknesses in the test data used for the program.

Mutation Testing is a **White Box Testing**.
Mutation testing can be applied to design models, specifications, databases, tests, and XML. It is a structural testing technique, which uses the structure of the code to guide the testing process. It can be described as the process of rewriting the source code in small ways in order to remove the redundancies in the source code.

CODE:-

```
#include<bits/stdc++.h>

#include<cstdio>

#include<iostream>

using namespace std;

int main(){



// Muation testing on triangle classification problem

ifstream myfile;

int cnt=0;

string line;
```

```
myfile.open("triangle.txt");

if (myfile.is_open()) {

            int e = 0, n = 0, p = 1;

            bool isStartProg;

            size_t found;

            while (getline(myfile, line)) {

                  if (line.empty())

                        continue;

                  if (!isStartProg) {

                        found = line.find("void main()");

                        if (found != string::npos) {

                              isStartProg = true;

                              e++;

                              n++;

                        cout << line << endl;

                        }

                  }

                  else {

                        if(line.find_first_of("//") != string::npos)

                              continue;

                        if(line.find("if") != string::npos)

                        {

                              cnt++;

                        }

                        e++;
```

```
                    n++;
                    cout << line << endl;
                }
        } }
        myfile.close();

cout<<"\nEnter test suit :";
int a[5],b[5],c[5];
string expected[5];
for(int i=0;i<5;i++)
cin>>a[i]>>b[i]>>c[i]>>expected[i];
cout<<"\n\nGiven test suit is\n";
cout<<"S.No  A  |  B   |  C   |  Expected Output   \n";
cout<<"----------------------------------------------\n";
for(int i=0;i<5;i++)
cout<<i+1<<"      "<<a[i]<<"   |"<<"   "<<b[i]<<"   |"<<"   "<<c[i]<<"   |"<<"   "<<expected[i]<<"
\n";
cout<<"\n\n\n Enter Number of mutations : ";
int num_mut;
cin>>num_mut;
int m[num_mut];
int mutation_killed=0;
for(int i=0;i<num_mut;i++)
{
```

```
cout<<"\nEnter serial number of mutant: ";

cin>>m[i];

cout<<"\nEnter line of mutation: ";

int line_mut;

cin>>line_mut;

string mutation;

cout<<"\nEnter mutation in form of string: ";

cin>>mutation;

bool check=false;

/// mutation 1

if(line_mut==13&&mutation=="a1")            //line=13

{

cout<<"\nAfter Mutation"<<" #"<<i+1<<"\n";

cout<<"------------------------------------------\n";

cout<<"  A \tB\tC\tExpected\tAfter Mutation  \n";

vector<string> after_mut(5);

for(int i=0;i<5;i++)

{

                if(a[i]>0)

                after_mut[i]="outofrange";

                else{

                int maximum=max(a[i],max(b[i],c[i]));

                   int minimum = min(a[i],min(b[i],c[i]));

                   int sum = a[i]+b[i]+c[i];

                   int middle = sum-maximum-minimum;
```

```
                    if(maximum>=(sum)-(maximum))
                    after_mut[i]="invalid";
                    else{
                              if((maximum*maximum)==((minimum*minimum)+(middle*middle)))
                              after_mut[i] = "right";
                              else if((maximum*maximum)<((minimum*minimum)+(middle*middle)))
                              after_mut[i] = "acute";
                              else
                              after_mut[i] = "obtuse";
                 }
}
                 if(after_mut[i]!=expected[i])
                 check=true;
                 cout<<"
"<<a[i]<<"\t"<<b[i]<<"\t"<<c[i]<<"\t"<<expected[i]<<"\t\t"<<after_mut[i]<<" \n";
}
if(check)
mutation_killed++;
if(check)
                 cout<<"mutant killed\n";
else
cout<<"Not Killed\n";
}
```

```
/// mutation 2

else if(line_mut==13&&mutation=="a5")                //line=13

{

cout<<"\nAfter Mutation"<<" #"<<i+1<<"\n";

cout<<"------------------------------------------\n";

cout<<"  A \tB\tC\tExpected\tAfter Mutation  \n";

vector<string> after_mut(5);


for(int i=0;i<5;i++)

{

                if(b[i]>0)

                after_mut[i]="outofrange";

                else{

                int maximum=max(a[i],max(b[i],c[i]));

                   int minimum = min(a[i],min(b[i],c[i]));

                   int sum = a[i]+b[i]+c[i];

                   int middle = sum-maximum-minimum;

                   if(maximum>=(sum)-(maximum))

                   after_mut[i]="invalid";

                   else{

                            if((maximum*maximum)==((minimum*minimum)+(middle*middle)))

                            after_mut[i] = "right";

                            else if((maximum*maximum)<((minimum*minimum)+(middle*middle)))

                            after_mut[i] = "acute";

                            else
```

```
                                after_mut[i] = "obtuse";

                }

}

                if(after_mut[i]!=expected[i])

                check=true;

                cout<<"
"<<a[i]<<"\t"<<b[i]<<"\t"<<c[i]<<"\t"<<expected[i]<<"\t\t"<<after_mut[i]<<" \n";


}

if(check)

mutation_killed++;

if(check)

                cout<<"mutant killed\n";

else

cout<<"Not Killed\n";

}


/// mutation 3

else if(line_mut==13&&mutation=="a9")                    //line=13

{

cout<<"\nAfter Mutation"<<" #"<<i+1<<"\n";

cout<<"-------------------------------------------\n";

cout<<"  A \tB\tC\tExpected\tAfter Mutation  \n";

vector<string> after_mut(5);

for(int i=0;i<5;i++)
```

```
{
          if(c[i]>0)
          after_mut[i]="outofrange";
          else{
            int maximum=max(a[i],max(b[i],c[i]));
            int minimum = min(a[i],min(b[i],c[i]));
            int sum = a[i]+b[i]+c[i];
            int middle = sum-maximum-minimum;
            if(maximum>=(sum)-(maximum))
            after_mut[i]="invalid";
            else{
                  if((maximum*maximum)==((minimum*minimum)+(middle*middle)))
                  after_mut[i] = "right";
                  else if((maximum*maximum)<((minimum*minimum)+(middle*middle)))
                  after_mut[i] = "acute";
                  else
                  after_mut[i] = "obtuse";
          }
}
          if(after_mut[i]!=expected[i])
          check=true;
          cout<<"
"<<a[i]<<"\t"<<b[i]<<"\t"<<c[i]<<"\t"<<expected[i]<<"\t\t"<<after_mut[i]<<" \n";


}
```

```
if(check)
mutation_killed++;
if(check)
                cout<<"mutant killed\n";
else
cout<<"Not Killed\n";
}

// mutation 4
else if(line_mut==13&&mutation=="a2")              //line 13
{
cout<<"\nAfter Mutation"<<" #"<<i+1<<"\n";
cout<<"------------------------------------------\n";
cout<<"  A \tB\tC\tExpected\tAfter Mutation  \n";
vector<string> after_mut(5);
for(int i=0;i<5;i++)
{
                if(a[i]>=100||a[i]<0)
                after_mut[i]="outofrange";
                else{
                   int maximum=max(a[i],max(b[i],c[i]));
                   int minimum = min(a[i],min(b[i],c[i]));
                   int sum = a[i]+b[i]+c[i];
                   int middle = sum-maximum-minimum;
                   if(maximum>=(sum)-(maximum))
```

```
                        after_mut[i]="invalid";
                        else{
                                    if((maximum*maximum)==((minimum*minimum)+(middle*middle)))
                                    after_mut[i] = "right";
                                    else if((maximum*maximum)<((minimum*minimum)+(middle*middle)))
                                    after_mut[i] = "acute";
                                    else
                                    after_mut[i] = "obtuse";
                        }
                    }
                    if(after_mut[i]!=expected[i])
                    check=true;
                    cout<<"
"<<a[i]<<"\t"<<b[i]<<"\t"<<c[i]<<"\t"<<expected[i]<<"\t\t"<<after_mut[i]<<" \n";


}
if(check)
mutation_killed++;
if(check)
                    cout<<"mutant killed\n";
else
cout<<"Not killed\n";
}
```

```
/// Mutation 5

else if(line_mut==14&&mutation=="a3")  // line 14

{

cout<<"\nAfter Mutation"<<" #"<<i+1<<"\n";

cout<<"-------------------------------------------\n";

cout<<"  A \tB\tC\tExpected\tAfter Mutation  \n";

vector<string> after_mut(5);

for(int i=0;i<5;i++)

{

                    if(expected[i]=="outofrange")

                    {

                        after_mut[i]=expected[i];

                        continue;

                    }

int maximum=max(a[i],max(b[i],c[i]));

int minimum = min(a[i],min(b[i],c[i]));

int sum = a[i]+b[i]+c[i];

int middle = sum-maximum-minimum;

                    if(maximum<minimum+middle)

                    after_mut[i]="invalid";

                    else{

                        after_mut[i]="obtuse";

                    }

                    if(after_mut[i]!=expected[i])

                    check=true;
                    cout<<"
"<<a[i]<<"\t"<<b[i]<<"\t"<<c[i]<<"\t"<<expected[i]<<"\t\t"<<after_mut[i]<<" \n";

}

if(check)

mutation_killed++;

if(check)

                    cout<<"mutant killed\n";

else

cout<<"Not Killed\n";

}


/// Mutation 6

else if(line_mut==28&&mutation=="a6")     // line 28

{

cout<<"\nAfter Mutation"<<" #"<<i+1<<"\n";

cout<<"------------------------------------------\n";

cout<<"  A \tB\tC\tExpected\tAfter Mutation  \n";

vector<string> after_mut(5);

for(int i=0;i<5;i++)

{

                    if(expected[i]=="right")

                    after_mut[i]="acute";

                    else{

                        after_mut[i]=expected[i];

                    }
                    if(after_mut[i]!=expected[i])

                    check=true;

                    cout<<"
"<<a[i]<<"\t"<<b[i]<<"\t"<<c[i]<<"\t"<<expected[i]<<"\t\t"<<after_mut[i]<<" \n";

}

if(check)

mutation_killed++;

if(check)

                    cout<<"mutant killed\n";

else

cout<<"Not Killed\n";

}


//// Mutation 7

else if(line_mut==35&&mutation=="a4")                 // line 35

{

cout<<"\nAfter Mutation"<<" #"<<i+1<<"\n";

cout<<"-------------------------------------------\n";

cout<<"  A \tB\tC\tExpected\tAfter Mutation  \n";

vector<string> after_mut(5);

for(int i=0;i<5;i++)

{

                    if(expected[i]=="invalid")

                    after_mut[i]="outofrange";

                    else{
```

```
                after_mut[i]=expected[i];
            }
            if(after_mut[i]!=expected[i])
            check=true;
            cout<<"
"<<a[i]<<"\t"<<b[i]<<"\t"<<c[i]<<"\t"<<expected[i]<<"\t\t"<<after_mut[i]<<" \n";
}
if(check)
mutation_killed++;
if(check)
            cout<<"mutant killed\n";
else
cout<<"Not Killed\n";
}


// Mutation 8
else if(line_mut==13&&mutation=="a7")           // line 13
{
cout<<"\nAfter Mutation"<<" #"<<i+1<<"\n";
cout<<"------------------------------------------\n";
cout<<"  A \tB\tC\tExpected\tAfter Mutation  \n";
vector<string> after_mut(5);
for(int i=0;i<5;i++)
{
            if(c[i]>=100||c[i]<=0||expected[i]=="outofrange")
```

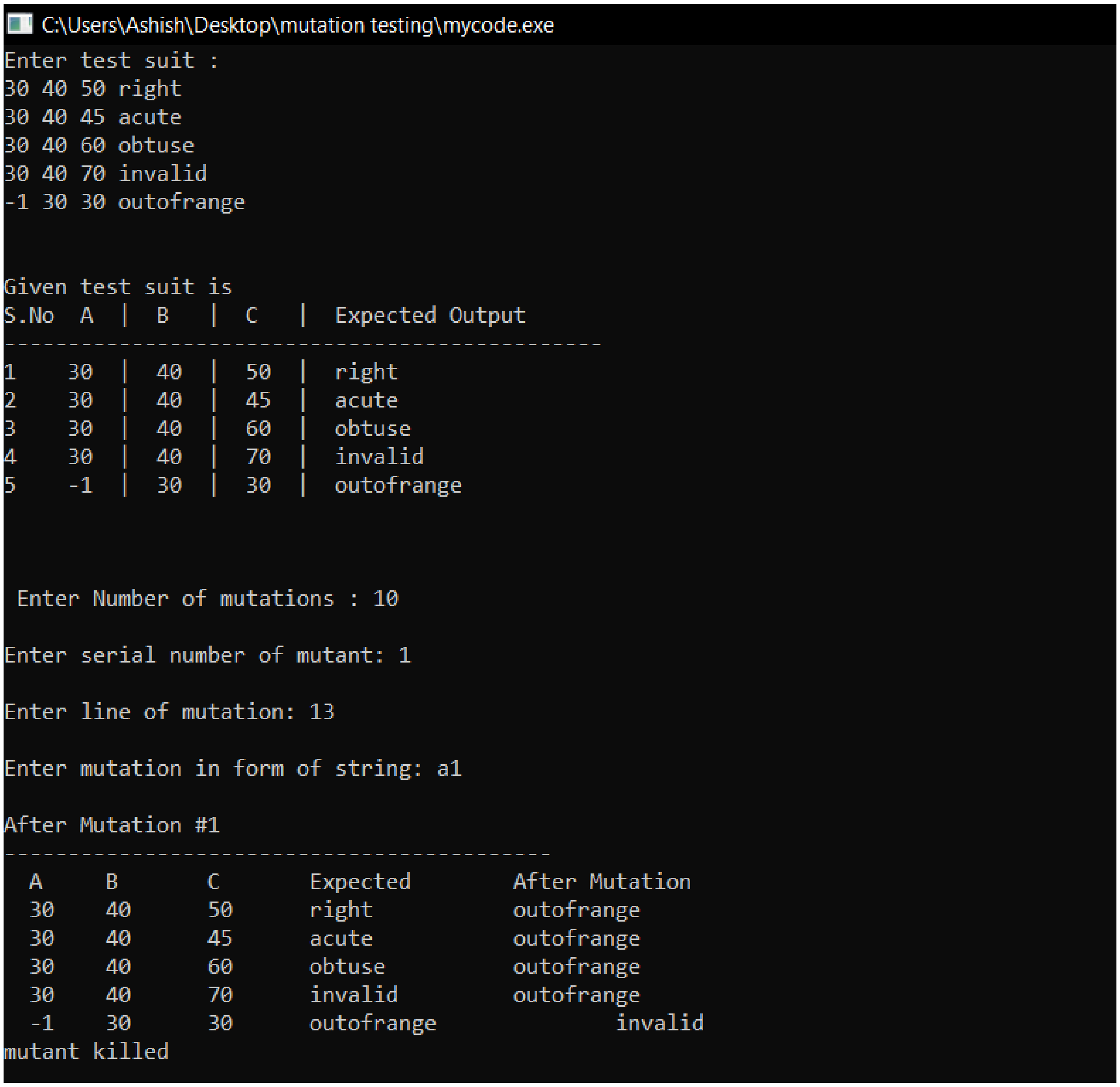
```
                    after_mut[i]="outofrange";
                    else{
                      int maximum=max(a[i],max(b[i],c[i]));
                      int minimum = min(a[i],min(b[i],c[i]));
                      int sum = a[i]+b[i]+c[i];
                      int middle = sum-maximum-minimum;
                      if(maximum>=(sum)-(maximum))
                      after_mut[i]="invalid";
                      else{
                              if((maximum*maximum)==((minimum*minimum)+(middle*middle)))
                              after_mut[i] = "right";
                              else if((maximum*maximum)<((minimum*minimum)+(middle*middle)))
                              after_mut[i] = "acute";
                              else
                              after_mut[i] = "obtuse";
                      }
                    }
                    if(after_mut[i]!=expected[i])
                    check=true;
                    cout<<"
"<<a[i]<<"\t"<<b[i]<<"\t"<<c[i]<<"\t"<<expected[i]<<"\t\t"<<after_mut[i]<<" \n";
}
if(check)
mutation_killed++;
if(check)
```

```
                    cout<<"mutant killed\n";
else
cout<<"Not Killed\n";
}


// Mutation 9
else if(line_mut==21&&mutation=="a4")                  // line 21
{
                    cout<<"\nAfter Mutation"<<" #"<<i+1<<"\n";
cout<<"------------------------------------------\n";
cout<<"  A \tB\tC\tExpected\tAfter Mutation  \n";
vector<string> after_mut(5);
for(int i=0;i<5;i++)
{
                    if(expected[i]!="invalid")
                    after_mut[i]="outofrange";
                    else{
                        after_mut[i]="obtuse";
                    }
                    if(after_mut[i]!=expected[i])
                    check=true;
                    cout<<"
"<<a[i]<<"\t"<<b[i]<<"\t"<<c[i]<<"\t"<<expected[i]<<"\t\t"<<after_mut[i]<<" \n";
}
if(check)
```

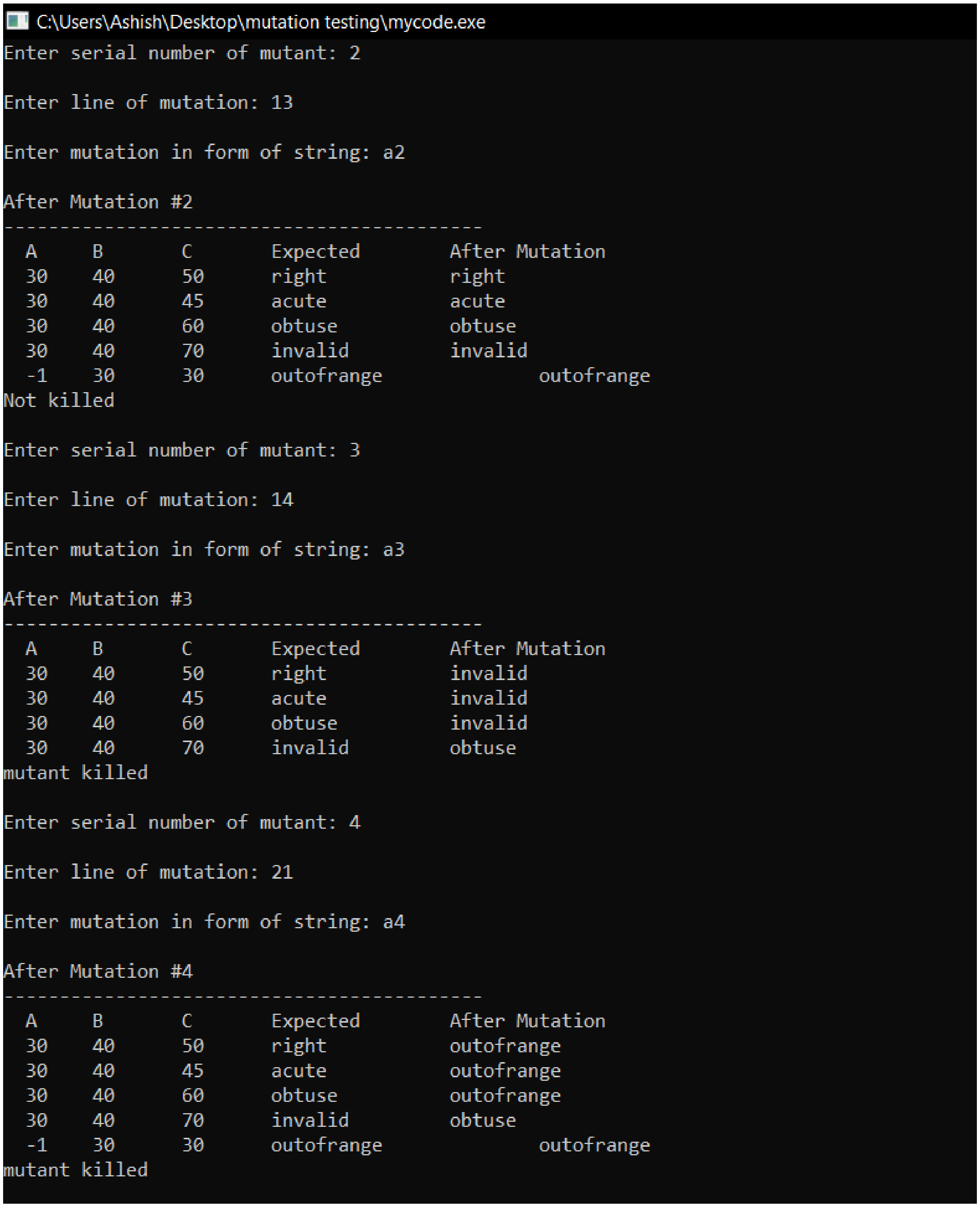
```
}

mutation_killed++;

if(check)

                    cout<<"mutant killed\n";

else

cout<<"Not Killed\n";

}


//Mutation 10

else if(line_mut==13&&mutation=="a8")                  //line 13

{

cout<<"\nAfter Mutation"<<" #"<<i+1<<"\n";

cout<<"-------------------------------------------\n";

cout<<"  A \tB\tC\tExpected\tAfter Mutation  \n";

vector<string> after_mut(5);

for(int i=0;i<5;i++)

{

                    if(b[i]!=0)

                    after_mut[i]=expected[i];

                    else

                    after_mut[i]="invalid";

                    if(after_mut[i]!=expected[i])

                    check=true;

                    cout<<"
"<<a[i]<<"\t"<<b[i]<<"\t"<<c[i]<<"\t"<<expected[i]<<"\t\t"<<after_mut[i]<<" \n";

}
```

```
if(check)

mutation_killed++;

if(check)

                    cout<<"mutant killed\n";

else

cout<<"Not Killed\n";

}

}

cout<<"mutant killed = "<<mutation_killed<<endl;

cout<<"Total number of mutations is 10\n";

float ans=mutation_killed/num_mut;

cout<<"\nSo, Mutation_Score: 7/10 =  "<<ans<<"\n";

return 0;

}
```

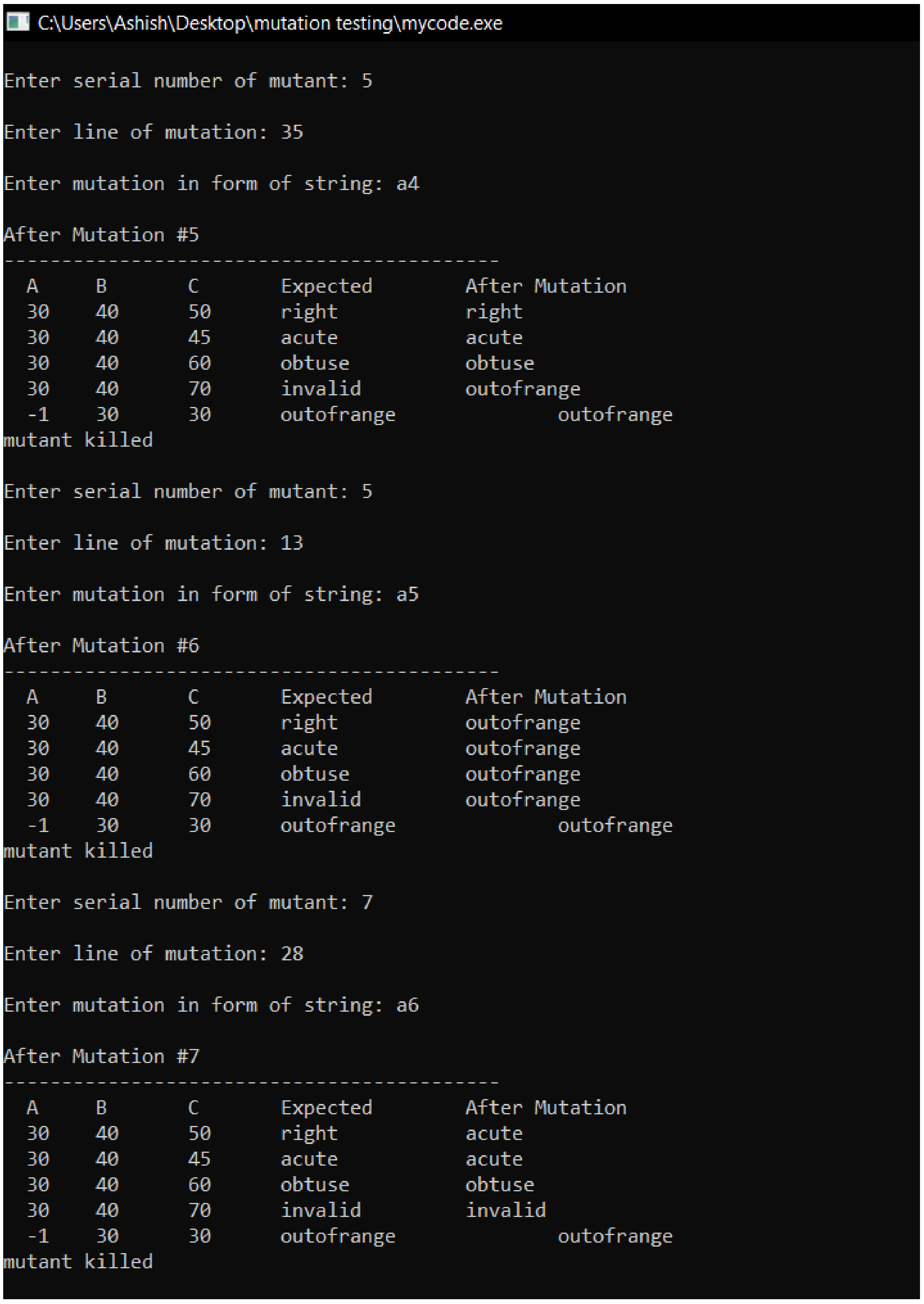
OUTPUT:-



```
C:\Users\Ashish\Desktop\mutation testing\mycode.exe
Enter test suit :
30 40 50 right
30 40 45 acute
30 40 60 obtuse
30 40 70 invalid
-1 30 30 outofrange


Given test suit is
S.No  A  |  B   |  C   |  Expected Output
----------------------------------------------
1    30  |  40  |  50  |  right
2    30  |  40  |  45  |  acute
3    30  |  40  |  60  |  obtuse
4    30  |  40  |  70  |  invalid
5    -1  |  30  |  30  |  outofrange



 Enter Number of mutations : 10

Enter serial number of mutant: 1

Enter line of mutation: 13

Enter mutation in form of string: a1

After Mutation #1
------------------------------------------
  A     B       C       Expected        After Mutation
  30    40      50      right           outofrange
  30    40      45      acute           outofrange
  30    40      60      obtuse          outofrange
  30    40      70      invalid         outofrange
  -1    30      30      outofrange              invalid
mutant killed
```

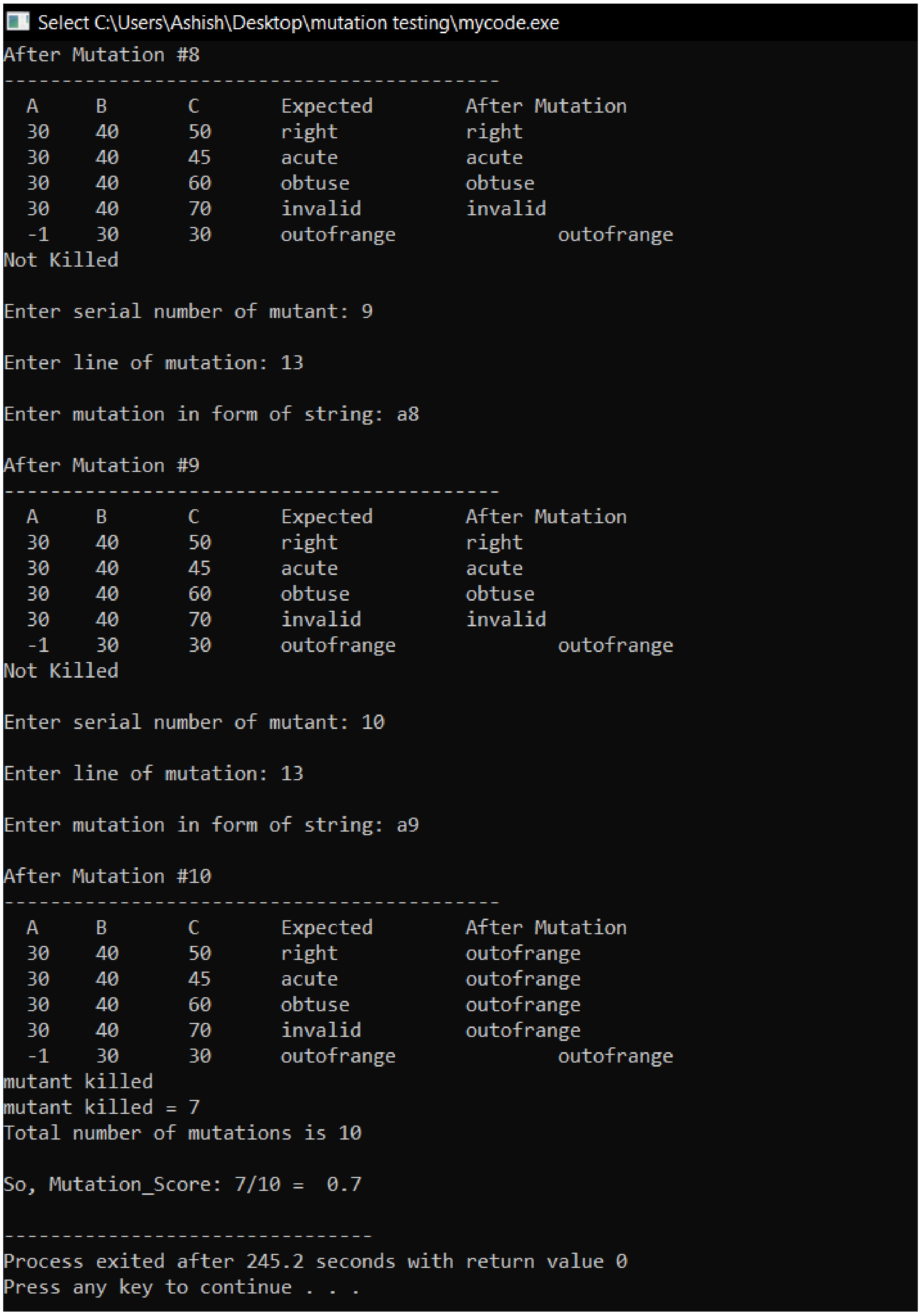

C:\Users\Ashish\Desktop\mutation testing\mycode.exe

```
Enter serial number of mutant: 2

Enter line of mutation: 13

Enter mutation in form of string: a2

After Mutation #2
-------------------------------------------
  A     B       C       Expected        After Mutation
  30    40      50      right           right
  30    40      45      acute           acute
  30    40      60      obtuse          obtuse
  30    40      70      invalid         invalid
  -1    30      30      outofrange              outofrange
Not killed

Enter serial number of mutant: 3

Enter line of mutation: 14

Enter mutation in form of string: a3

After Mutation #3
-------------------------------------------
  A     B       C       Expected        After Mutation
  30    40      50      right           invalid
  30    40      45      acute           invalid
  30    40      60      obtuse          invalid
  30    40      70      invalid         obtuse
mutant killed

Enter serial number of mutant: 4

Enter line of mutation: 21

Enter mutation in form of string: a4

After Mutation #4
-------------------------------------------
  A     B       C       Expected        After Mutation
  30    40      50      right           outofrange
  30    40      45      acute           outofrange
  30    40      60      obtuse          outofrange
  30    40      70      invalid         obtuse
  -1    30      30      outofrange              outofrange
mutant killed
```



C:\Users\Ashish\Desktop\mutation testing\mycode.exe

```
Enter serial number of mutant: 5

Enter line of mutation: 35

Enter mutation in form of string: a4

After Mutation #5
-------------------------------------------
  A     B       C       Expected        After Mutation
  30    40      50      right           right
  30    40      45      acute           acute
  30    40      60      obtuse          obtuse
  30    40      70      invalid         outofrange
  -1    30      30      outofrange              outofrange
mutant killed

Enter serial number of mutant: 5

Enter line of mutation: 13

Enter mutation in form of string: a5

After Mutation #6
-------------------------------------------
  A     B       C       Expected        After Mutation
  30    40      50      right           outofrange
  30    40      45      acute           outofrange
  30    40      60      obtuse          outofrange
  30    40      70      invalid         outofrange
  -1    30      30      outofrange              outofrange
mutant killed

Enter serial number of mutant: 7

Enter line of mutation: 28

Enter mutation in form of string: a6

After Mutation #7
-------------------------------------------
  A     B       C       Expected        After Mutation
  30    40      50      right           acute
  30    40      45      acute           acute
  30    40      60      obtuse          obtuse
  30    40      70      invalid         invalid
  -1    30      30      outofrange              outofrange
mutant killed
```



```
Select C:\Users\Ashish\Desktop\mutation testing\mycode.exe
After Mutation #8
-------------------------------------------
  A     B       C       Expected        After Mutation
  30    40      50      right           right
  30    40      45      acute           acute
  30    40      60      obtuse          obtuse
  30    40      70      invalid         invalid
  -1    30      30      outofrange              outofrange
Not Killed

Enter serial number of mutant: 9

Enter line of mutation: 13

Enter mutation in form of string: a8

After Mutation #9
-------------------------------------------
  A     B       C       Expected        After Mutation
  30    40      50      right           right
  30    40      45      acute           acute
  30    40      60      obtuse          obtuse
  30    40      70      invalid         invalid
  -1    30      30      outofrange              outofrange
Not Killed

Enter serial number of mutant: 10

Enter line of mutation: 13

Enter mutation in form of string: a9

After Mutation #10
-------------------------------------------
  A     B       C       Expected        After Mutation
  30    40      50      right           outofrange
  30    40      45      acute           outofrange
  30    40      60      obtuse          outofrange
  30    40      70      invalid         outofrange
  -1    30      30      outofrange              outofrange
mutant killed
mutant killed = 7
Total number of mutations is 10

So, Mutation_Score: 7/10 =  0.7

--------------------------------
Process exited after 245.2 seconds with return value 0
Press any key to continue . . .
```