Software Testing
Assignment 2

Submitted by :
Shiv Kumar
2016UIT2563
Semester 8
IT2

Experiment 7 : Write any program with the following conditions:

1. **LOC should be at-least 30.**
2. **There should be use of at-least 1 conditional statements (if-else, switch case).**
3. **There should be use of at-least 1 looping structure (do while, while, for).**

Perform data flow testing for the above program showing all the steps.

CODE:

```
int main() {

long int N;
long int s = 0;
long int p = 0;
cin>>N;
long int a[N];
long int i=0;
while(i>a[i]){
        cin>>a[i];
        if(a[i] < 0 || a[i] > 100) {
                continue;
        }
        i++;
}
i=0;
while(i<N){
        int X = a[i];
        int count = 0;
        while(X) {
                if(X%2==1) {
                count++;
                }
        X/=2;
        }
        s+= count;
        p*= count;
        i++;
}

cout<<s<<endl;
cout<<p<<endl;

}
```

CONTROL FLOW GRAPH

DEFINITION AND USE OF ALL VARIABLES

S.NO	VARIABLE	DEFINE	USE
1	s	3,25	25, 29
2	p	4,26	26, 30
3	N	5	6, 8, 16
4	i	7, 13, 15, 27	8, 9, 10, 13, 16, 17, 27
5	X	17, 23	19, 20, 23
6	count	18, 21	21, 25, 26

DEFINING ALL DU PATHS

S.NO	VARIABLE	PATHS
1	s	1. (3, 25, s) 2. (3, 29, s) 3. (25,25,s) 4. (25,29,s)
2	p	1. (4,26,p) 2. (4,30,p) 3. (26,26,p) 4. (26,30,p)
3	N	1. (5,6,N) 2. (5,8,N) 3. (5,16,N)
4	i	1. (7,8,i) 2. (7,9,i) 3. (7,10,i) 4. (7,13,i) 5. (13,8,i) 6. (13,9,i) 7. (13,10,i) 8. (13,13,i) 9. (15,16,i) 10. (15,17,i) 11. (15,27,i) 12. (27,16,i) 13. (27,17,i) 14. (27,27,i)
5	X	1. (17,19,X) 2. (17,20,X) 3. (17,23,X) 4. (23,19,X) 5. (23,20,X) 6. (23,23,X)

S.NO	VARIABLE	PATHS
6	count	1. (18,21,count) 2. (18,25,count) 3. (18,26,count) 4. (21,21,count) 5. (21,25,count) 6. (21,26,count)

TESTING

i) FIND ALL DEFINITION COVERAGE:

1. (3, 25, sumOfSetBits)
2. (4,26,productOfSetBits)
3. (5,6,N)
4. (7,(8,f),i)
5. (13,(8,f),i)
6. (15,(16,f),i)
7. (17,(19,f),X)
8. (21,21,count)
9. (23,(19,f),X)
10. (25,29,sumOfSetBits)
11. (26,30,productOfSetBits)
12. (27,(16,f),i)

ii) FIND ALL C-USE COVERAGE:

1. (3, 25, sumOfSetBits)
2. (3, 29, sumOfSetBits)
3. (4,26,productOfSetBits)
4. (4,30,productOfSetBits)
5. (5,6,N)
6. (5,8,N)
7. (5,16,N)
8. (7,9,i)
9. (7,13,i)
10. (13,9,i)
11. (13,13,i)
12. (15,17,i)
13. (15,27,i)
14. (17,23,X)
15. (18,21,count)
16. (18,25,count)
17. (18,26,count)
18. (21,21,count)
19. (21,25,count)
20. (21,26,count)
21. (23,23,X)
22. (25,25,sumOfSetBits)
23. (25,29,sumOfSetBits)

24.(26,26,productOfSetBits)
25.(26,30,productOfSetBits)
26.(27,17,i) 27.(27,27,i)

iii) FIND ALL C-USE SOME P-USE COVERAGE:

1. (3, 25, sumOfSetBits)
2. (3, 29, sumOfSetBits)
3. (4,26,productOfSetBits)
4. (4,30,productOfSetBits)
5. (5,6,N)
6. (5,8,N)
7. (5,16,N)
8. (7,9,i)
9. (7,13,i)
10.(13,9,i)
11.(13,13,i)
12.(15,17,i)
13.(15,27,i)
14.(17,23,X)
15.(18,21,count)
16.(18,25,count)
17.(18,26,count)
18.(21,21,count)
19.(21,25,count)
20.(21,26,count)
21.(23,23,X)
22.(25,25,sumOfSetBits)
23.(25,29,sumOfSetBits)
24.(26,26,productOfSetBits)
25.(26,30,productOfSetBits)
26.(27,17,i)
27.(27,27,i)

iv) FIND ALL P-USE SOME C-USE COVERAGE:

1. (3, 25, sumOfSetBits)
2. (4,26,productOfSetBits)
3. (5,6,N)
4. (7,(8,f),i)
5. (7,(8,t),i)
6. (7,(10,f),i)
7. (7,(10,t),i)
8. (13,(8,f),i)
9. (13,(8,t),i)
10.(13,(10,f),i)
11.(13,(10,t),i)
12.(15,(16,f),i)
13.(15,(16,t),i)
14.(17,(19,f),X)
15.(17,(19,t),X)

16.(17,(20,f),X)
17.(17,(20,t),X)
18.(18,21,count)
19.(21,21,count)
20.(23,(19,f),X)
21.(23,(19,t),X)
22.(23,(20,f),X)
23.(23,(20,t),X)
24.(25,29,sumOfSetBits)
25.(26,30,productOfSetBits)
26.(27,(16,f),i)
27.(27,(16,t),i)

v) DESIGN THE TEST-CASE

Let the possible input be: 1 101 6.

Path:
1-2-3-4-5-6-7-8-9-10-11-12-8-9-10-13-14-8-15-16-17-18-19-20-23-24-19-20-21-22-23-24-19-20-21-22-23-24-19-25-26-27-28-16-29-30-31

This covers all the DU Paths and all the Definitions.

Experiment 8 : Write any program with the following conditions:

1. **LOC should be at-least 30.**
2. **There should be use of at-least 1 conditional statements (if-else, switch case).**
3. **There should be use of at-least 1 looping structure (do while, while, for).**

Perform slice based testing for the above program showing all the steps.

CODE:

```
int main() {

long int N;
long int s = 0;
long int p = 0;
cin>>N;
long int a[N];
long int i=0;
while(i>a[i]){
        cin>>a[i];
        if(a[i] < 0 || a[i] > 100) {
                continue;
        }
        i++;
}
i=0;
while(i<N){
        int X = a[i];
        int count = 0;
        while(X) {
                if(X%2==1) {
                count++;
                }
        X/=2;
        }
        s+= count;
        p*= count;
        i++;
}

cout<<s<<endl;
cout<<p<<endl;

}
```

CONTROL FLOW GRAPH

MAKE THE POSSIBLE SLICES FOR EACH VARIABLE

S.NO.	VARIABLE	SLICES
1	N	1. S(N,5) = [1-2, 5, 31] 2. S(N,31) = [1-2, 5-8, 13-14, 15-16, 27-28, 31]
2	s	1. S(sumOfSetBits,3) = [1, 3, 31] 2. S(sumOfSetBits,25) = [1-3, 5-9, 13-25, 27-28, 31] 3. S(sumOfSetBits,29) = [1-3, 5-9, 13-25, 27-29, 31] 4. S(sumOfSetBits,31) = [1-3, 5-9, 13-25, 27-29, 31]
3	p	1. S(productOfSetBits,4) = [1, 4, 31] 2. S(productOfSetBits,26) = [1-2, 4-9, 13-24, 26-28, 31] 3. S(productOfSetBits,30) = [1-2, 4-9, 13-24, 26-28, 30-31] 4. S(sumOfSetBits,31) = [1-2, 4-9, 13-24, 26-28, 30-31]
4	arr	1. S(arr,9) = [1-2, 5-9, 13-14, 31] 2. S(arr,31) = [1-2, 5-17, 28, 31]
5	i	1. S(i,7) = [1, 7, 31] 2. S(i,13) = [1, 7-10, 12-14, 31] 3. S(i,15) = [1, 7-10, 12-14, 15, 31] 4. S(i,27) = [1, 7-10, 12-14, 15-17, 27-28, 31] 5. S(i,31) = [1, 7-10, 12-14, 15-17, 27-28, 31]
6	X	1. S(X,17) = [1-2, 5-9, 13-17, 27-28, 31] 2. S(X,23) = [1-2, 5-9, 13-17, 19-20, 22-24, 27-28, 31] 3. S(X,31) = [1-2, 5-9, 13-17, 19-20, 22-24, 27-28, 31]
7	count	1. S(count,18) = [1-2, 5-9, 13-18, 27-28, 31] 2. S(count,21) = [1-2, 5-9, 13-22, 24, 27-28, 31] 3. S(count,31) = [1-2, 5-9, 13-22, 24-28, 31]

DESIGN THE TEST CASES FOR EACH SLICE

S.NO.	SLICE COVERED	N	arr	EXPECTED O/P
1	S(N, 5)	1	-	-
2	S(N, 31)	1	-	-
3	S(s,3)	-	-	-
4	S(s,25)	2	[6,4]	-
5	S(s,29)	2	[6,4]	3
6	S(s,31)	2	[6,4]	3
7	S(p,4)	-	-	-
8	S(p,26)	2	[6,4]	-
9	S(p,30)	2	[6,4]	2
10	S(p,31)	2	[6,4]	2

S.NO.	SLICE COVERED	N	arr	EXPECTED O/P
11	S(arr,9)	2	[6,4]	-
12	S(arr,31)	2	[6,4]	-
13	S(i,7)	-	-	-
14	S(i,13)	2	[6,4]	-
15	S(i,15)	2	[6,4]	-
16	S(i,27)	2	[6,4]	-
17	S(i,31)	2	[6,4]	-
18	S(X,17)	2	[6,4]	-
19	S(X,23)	2	[6,4]	-
20	S(X,31)	2	[6,4]	-
21	S(count,18)	2	[6,4]	-
22	S(count, 21)	2	[6,4]	-
23	S(count, 31)	2	[6,4]	-