[cover sheet, phase 1 and full assignment]

3

ASSIGNMENT 3

**CYCLOMATIC COMPLEXITY, PHASE 1**

CSE 6329 -- SOFTWARE MEASUREMENT AND QUALITY ENGINEERING

Professor Dennis J. Frailey

**Spring, 2018**

|  |  |
| --- | --- |
| NAME:**1)Karthik Somanahalli Muralidhara** | STUDENT ID NUMBER : **1001520125** |
| **2)Subrahmanya Basavapatna Nagaraja Rao** | **1001540208** |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | | **Cyclomatic Complexity Calculation** | | | | |
| **Program** | **Arcs** | | **Nodes** | **C (Number of Separate Flowgraphs)** | **Arcs - Nodes** | **Arcs – Nodes + 2C (Cyclomatic Complexity)** |
| Main | 21 | | 19 | 1 | 2 | 4 |
| Function | 53 | | 42 | 1 | 11 | 13 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Grading Comments (student – do not write inside this box)** | | | | |
| **Phase 1** | | | | |
| **<total goes here>** | **/10** | **Flowgraph (main)** | | **Comments from Grader** |
| **/5** | **Complexity (main)** | |
| **/10** | **Program (function)** | |
| **/10** | **Flowgraph (function)** | |
| **/ 5** | **Complexity (function)** | |
| **Phase 2** | | | | |
| **<total goes here>** | **<name>** | | First Program Reviewed | |
| **/ 5** | | **Errors** |  |
| **/ 5** | | **Cyclomatic Complexity** |
| **/ 10** | | **Flowgraph** |
| **<total goes here>** | **<name>** | | Second Program Reviewed | |
| **/ 5** | | **Errors** |  |
| **/ 5** | | **Cyclomatic Complexity** |
| **/ 10** | | **Flowgraph** |
| **<total goes here>** | **<name>** | | Third Program Reviewed | |
| **/ 5** | | **Errors** |  |
| **/ 5** | | **Cyclomatic Complexity** |
| **/ 10** | | **Flowgraph** |
| **<total>** | **Grand Total** | | |  |

**Place flowgraphs on next page(s)1) Main Program**

The C code for the Main program is given below :-

1

2

3 #include<stdio.h>

4 #include<math.h>

5

6 //Declare Stats () function

7 \_Bool Stats (int size, int inArray[ ], int outArray[ ],int \*mode, float \*mean, float \*median, float \*variance, float \*stdev)

8

9 int main( )

10 {

11 // Initialize values of the outputs

12 int mode = 0.0;

13 float mean = 0.0;

14 float median = 0.0;

15 float variance = 0.0;

16 float stdev = 0.0;

17 \_Bool inputValidityFlag = 0;

18 int outArray[100], inArray[100];

19

20 int i; /\* Array index \*/

21

22 int size = 0;

23

24

25 //Get the size of data in the array

26

27 printf("Enter size of data set\n");

28 scanf ("%d", &size);

29 printf("\n# elements = %d", size);

30

31 //Get input

32

33 for(i=0; i <= size-1; i++)

34 {

35 printf("\nEnter data item: \n");

36 scanf("%d", &inArray[i]);

37 }

38

39 inputValidityFlag = Stats (size, inArray, outArray, &mode, &mean, &median, &variance, &stdev);

40 printf("\nValidityFlag = %d\n", inputValidityFlag);

41

42 if (inputValidityFlag)

43 {

44 for (i = 0; i <= size-1; i++)

45 {

46 printf("\ninArray[%d] = %10.2d outArray[%d] = %10.2d", i, inArray[i], i, outArray[i]);

47 }

48 printf("\nmode = %10.2d\n", mode);

49 printf("\nmean = %10.2f", mean);

50 printf("\nmedian = %10.2f", median);

51 printf("\nvariance = %10.2f", variance);

52 printf("\nstand deviation = %10.2\nf", stdev);

53

54

55 }

56 else

57 {

58 printf("\nInput is invalid; outputs are not displayed\n");

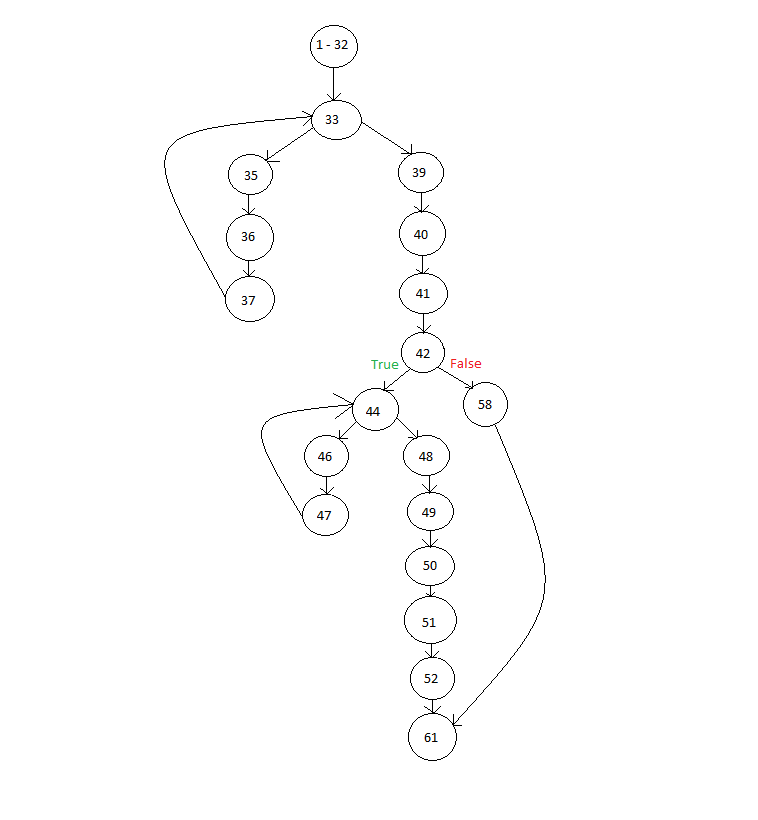
59 }

60

61 return 1;

62 }

The Control Flow Graph for Main program is displayed in the next page :-



Here , Cyclomatic Complexity is = e-n+2

= 21-19+2 = 4

Therefore **Cyclomatic Compexity** of this Main Program is **4**

**2) Stats Function**

The C code for the Stats Function is given below :-

1 Bool Stats (int size, int inArray[], int outArray[],int \*mode, float \*mean, float \*median, float \*variance, float \*stdev)

2 {

3 int i,j, temp, count = 0;

4 int T[size];

5 float x[8];

6 float s=0, s1=0; /\* Where s is the sum variable initialized to 0 and s1 is another sum variable \*/

7

8 /\* Sort the Array \*/

9

10 for(i=0; i<size; i++)

11 {

12 for(j=0; j<size-1; j++)

13 {

14 if(inArray[j]>inArray[j+1])

15 {

16 temp = inArray[j+1]; /\* Where temp is a temporary variable \*/

17 inArray[j+1] = inArray[j];

18 inArray[j] = temp;

19 }

20 temp = outArray[j];

21 }

22 }

23

24 /\* Compute Mean \*/

25

26 for (i = 0; i < size; i++)

27 {

28 s = s + x[i];

29 }

30 mean = s/(float)size; return mean;

31

32

33 /\* Compute median \*/

34

35 for(i = 0; i < size; i++)

36 for(j = i+1; j < size; j++)

37 {

38 if(inArray[i]>inArray[j])

39 {

40 temp=inArray[j];

41 inArray[j] = inArray[i];

42 inArray[i] = temp;

43 }

44 }

45 if(size%2==0)

46 {

47 median = (inArray[size/2]+inArray[total/2-1])/2;

48 }

49 else

50 {

51 median = inArray[size/2];

52 }

53 return median;

54

55 /\* Compute Mode \*/

56

57 for (i = 0; i < size; i++)

58 {

59 T[inArray[i]]++; /\* Where T[] is used to tally between current mode value and the total size of the array \*/

60 }

61 mode=0;

62

63 for (j = 0; j < size; j++)

64 {

65 if (T[j] > Count)

66 {

67 Count = T[j];

68 mode = j;

69 }

70 }

71 return mode;

72

73 /\* Compute variance and standard deviation \*/

74

75 for (i = 0; i < size; i++)

76 {

77 s1 = s1 + pow((x[i] - mean), 2);

78 }

79 variance = s1 / (float)size;

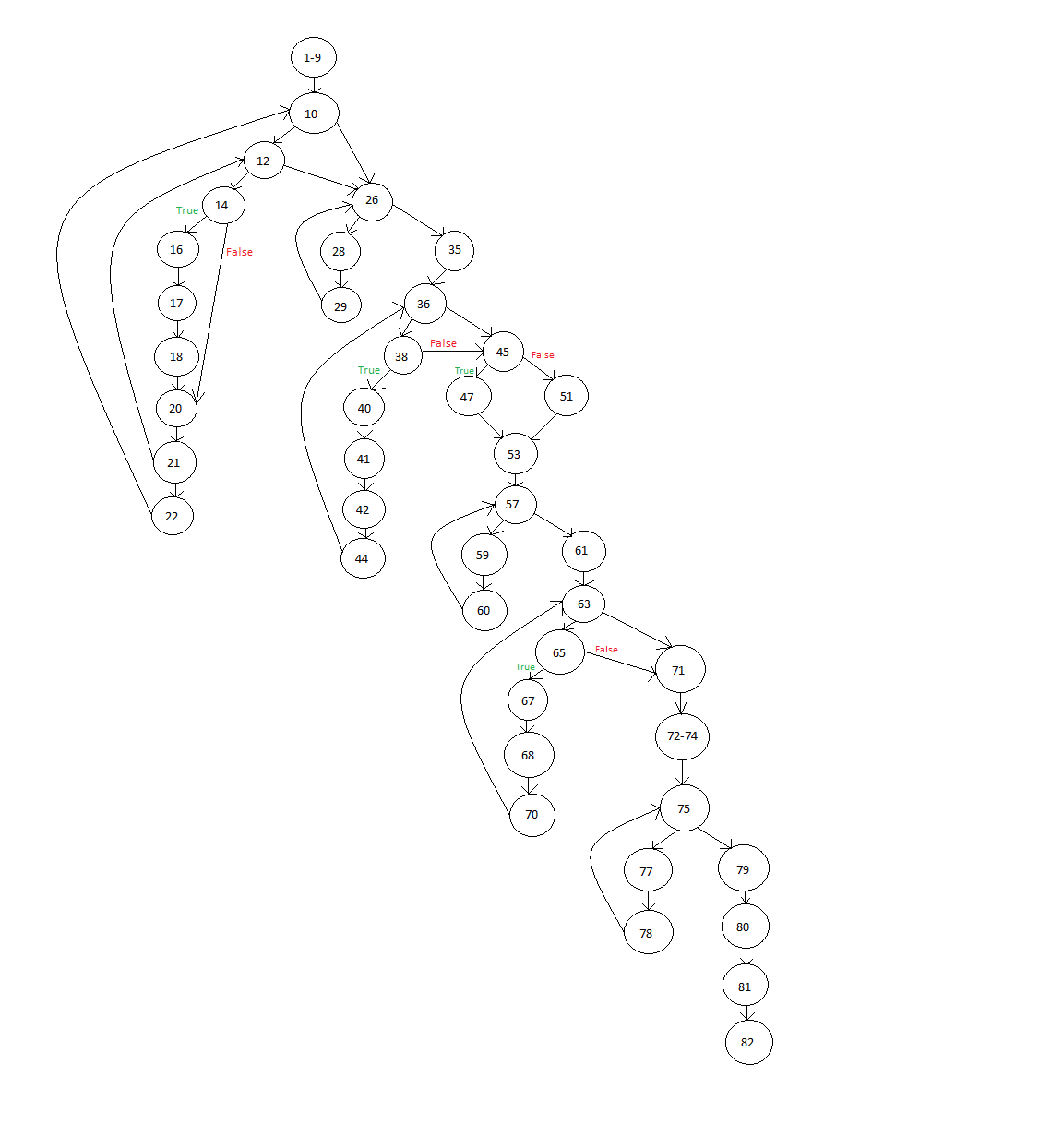
80 stdev = sqrt(variance);

81 return variance;

82 return stdev;

83 }

The Control Flow Graph for the Stats program is displayed in the next page:-



Here , Cyclomatic Complexity is = e-n+2

= 53-42+2 = 13

Therefore **Cyclomatic Compexity** of the Stats Program is **13**

**Note :-**  The predicate nodes are numbered as the sequence of line numbers, with respect to each line of code.

Moreover, Cyclomatic Complexity can also be determined as the number of closed regions + 1 i.e. There are 3 closed regions in the CFG of the main program, Thus the cyclomatic complexity is 4. Similarly, There are 12 closed regions of the CFG of the Stats Function, Thus the cyclomatic complexity is 13.