ECS505U Software Engineering

Test Report Document

Please write the test results of each test case using into the following tables. **You can enter as many test cases execution results as you want, however, you might lose marks if you provide 20 test cases for a section in which 5 test cases gives 100% coverage. Your task as a software engineer is to find minimum number of test cases that covers the code.**

If the number of rows in each table is not be enough simply add more rows. Document any defects discovered during testing. When finished save the file as pdf and submit.

1. Black Box

You can add more rows to both tables in this section.

Equivalence partitions:

Partition	exponent
1	int max >= exp >=1024 and exp is even
2	int max >= exp >=1024 and exp is odd
3	1023
4	0
5	normal range, even (e.g. 2)
6	normal range, odd (e.g. 3)
7	−1074
8	<= −1075 and exponent is even
9	<= −1075 and exponent is odd
10	1
11	-1

Test cases

exponent	Expected output	Observed output
Int max	-infinity	-infinity
Int max-1	+infinity	+infinity
1023	- Double Max (Adjusted Double Max)	- Double Max (Adjusted Double Max)
2 to 1022	$(-2)^{exponent}$	Ran $(-2)^{exponent}$ for inputs [2, 3, 5, 100, 1022]
1	-2.0	-2.0
0	1.0	1.0
-1	-0.5	-0.5
-2 to -1073	$(-2)^{exponent}$	Ran $(-2)^{exponent}$ for inputs [-1, -2, -5, -100, -1073] E.g., for -2 it gave -0.25
-1074	Double.MIN_VALUE	Double.MIN_VALUE
Int min	0.0	0.0
Int min + 1	-0.0	-0.0

Faults revealed:

When the function was run with the input -2 the expected output is supposed to be the output of $(-2)^{exponent}$, in this case $(-2)^{-2}$ or 0.25 but it returned -0.25.

2. Branch and Path Coverage

Branch Coverage

B1 : if(length > 20)
B2: if(length > 1)
B3: if(length == 0)

Inputs	Covered Branches
Length = 30	B1
Length = 20	B1,2
Length = 10	B1,2
Length = 1	B1,2,3
Length = 0	B1,2,3

Coverage score: 100%

Fault revealed: if the length = 1 nothing will be returned.

Path Coverage

Paths:

	Feasible				Infeasible					
First if	T	F	F	F	T	T	T	T	F	F
Second if	N/A	T	F	F	T	F	T/F	T/F	T	F
Third if	N/A	N/A	T	F	N/A	N/A	T	F	T	F

Test cases:

Inputs	Covered Paths
Length = 25	B1 – True
Length = 10	B1 – FALSE, B2 – TRUE
Length = 1	B1 – FALSE, B2 – FALSE, B3 -- FALSE
Length = 0	B1 – FALSE, B2 – FALSE, B3 -- TRUE

Coverage score: 100%

Faults revealed: Input array of type char with a length of 1 will result in unexpected / unwanted behavior

3. Challenging Branch Coverage (optional unmarked)

Inputs	Branches covered										
	(a > d && a < c + b && c < d)				else		while(z<10)			(a < b && b < c) || (a < c && b > d)	else
	a+c = 12	a+c = 8	a+c = 3	Def.	c > b && a < d	else	d < c	b == a	b == d		

Coverage score:

4. Test Driven Development

Copy and paste your code here.
Did your code pass all tests? YES
Code:

```
public class TemperatureConverter {
    public static double CELSIUS_MIN = -273.15;
    public static double CELSIUS_MAX = 5500000000000.0;
    public static double FAHRENHEIT_MIN =  -459.67;
    public static double FAHRENHEIT_MAX = 9900000000032.0;
    public static double KELVIN_MIN = 0;
    public static double KELVIN_MAX = 5500000000273.15;

    public static double F2K(double tempF) {
        if (tempF < FAHRENHEIT_MIN || tempF > FAHRENHEIT_MAX) {
            throw new IllegalArgumentException("Temperature in Fahrenheit is out of
bounds.");
        } else {
            return (tempF + 459.67) / 1.8;
        }
    }

    public static double F2C(double tempF) {
        if (tempF < FAHRENHEIT_MIN || tempF > FAHRENHEIT_MAX) {
            throw new IllegalArgumentException("Temperature in Fahrenheit is out of
bounds.");
        } else {
            return (tempF - 32) / 1.8;
        }
    }

    public  static double C2F(double tempC) {
        if (tempC < CELSIUS_MIN || tempC > CELSIUS_MAX) {
            throw new IllegalArgumentException("Temperature in Celsius is out of
bounds.");
        } else {
            return (tempC * 1.8) + 32;
        }
    }

    public  static double C2K(double tempC) {
        if (tempC < CELSIUS_MIN || tempC > CELSIUS_MAX) {
            throw new IllegalArgumentException("Temperature in Celsius is out of
bounds.");
        } else {
            return tempC + 273.15;
        }
    }

    public static double K2F(double tempK) {
        if (tempK < KELVIN_MIN || tempK > KELVIN_MAX) {
            throw new IllegalArgumentException("Temperature in Kelvin is out of
bounds.");
        } else {
            return (tempK * 1.8) - 459.67;
        }
    }

    public  static double K2C(double tempK) {
        if (tempK < KELVIN_MIN || tempK > KELVIN_MAX) {
            throw new IllegalArgumentException("Temperature in Kelvin is out of
bounds.");
        } else {
            return tempK - 273.15;
        }
    }
}
```