Assignment 1 Rubric

- Each question is of 10 marks (including the bonus)
- The **yardstick** will be the basis for giving out marks for any particular component.
- Each **component has associated marks** out of 10, which has been further **granulated as per the rationale** been provided.
- Bonus marks would be awarded if and only if the student has scored >=80% of the total marks.
- The Bonus marks would be 10% of the total marks.
- Question number 6 **OR** 7 is MANDATORY, if either of these is absent, 10% of the total marks would be deducted.
- The rubric is **binary.** Marks would only be awarded for a component, if it has been satisfactorily done, otherwise 0. [For example, if any component has 0.5 marks associated with it, then the student should be given full 0.5 marks, otherwise 0 for that component. This has to be followed for each of the components.]

Q1

Yardstick	Marks	Rationale
Code runs without error on students system	2	The program should execute at least once without errors.
Explainability	2	Student is able to briefly explain the question and required output.
Handling Patterns	5 (1 per pattern)	Student has handled all the pattern separately.
Code is able to run on any value of n , such that 1<= n <= 9	1 (0.2 per pattern)	

Q2

Yardstick	Marks	Rationale
Input values and print menu for user	2	User input for ‘n’ students, shape being 2D or 3D, printing menu for shapes of that category and then taking serial number input for shape
Perimeters, Areas and Volumes of different geometries	6	12 shapes in total: 0.5 mark each shape For 2D shape, 0.25 mark each for perimeter and area functions for each shape For 3D shape, 0.25 mark each for areas and volume functions for each shape
Testing	2	5 test cases, including 3 3D shapes, 2 2D shapes, 0.4 marks for each test case, they should be taking inputs for each shape picked, show menus, print correct values for ‘n’ students

Sample 2D test case

3
2D
1
3
2
1 4
5
7

Output

Menu:
1. Square input: side s
2. Rectangle input: length l, breadth b
3. Rhombus input: side a, diagonals d1,d2
4. Parallelogram input: length l, breadth b, height h

5. Circle input: radius r
6. Cube input: side s
7. Cuboid input: length l, breadth b, height h
8. Right circular cone: slant height l, radius r, height h
9. Hemisphere input: radius r
10. Sphere input: radius r
11. Solid cylinder input: radius r, height h
12. Hollow cylinder input: Radii R1, R2, height h

Square:
Perimeter=12
Area=9

Menu:
1. Square input: side s
2. Rectangle input: length l, breadth b
3. Rhombus input: side a, diagonals d1,d2
4. Parallelogram input: length l, breadth b, height h
5. Circle input: radius r
6. Cube input: side s
7. Cuboid input: length l, breadth b, height h
8. Right circular cone: slant height l, radius r, height h
9. Hemisphere input: radius r
10. Sphere input: radius r
11. Solid cylinder input: radius r, height h
12. Hollow cylinder input: Radii R1, R2, height h

Rectangle:
Perimeter=10
Area=4

Menu:
1. Square input: side s
2. Rectangle input: length l, breadth b
3. Rhombus input: side a, diagonals d1,d2
4. Parallelogram input: length l, breadth b, height h
5. Circle input: radius r
6. Cube input: side s
7. Cuboid input: length l, breadth b, height h
8. Right circular cone: slant height l, radius r, height h
9. Hemisphere input: radius r
10. Sphere input: radius r
11. Solid cylinder input: radius r, height h

12. Hollow cylinder input: Radii R1, R2, height h

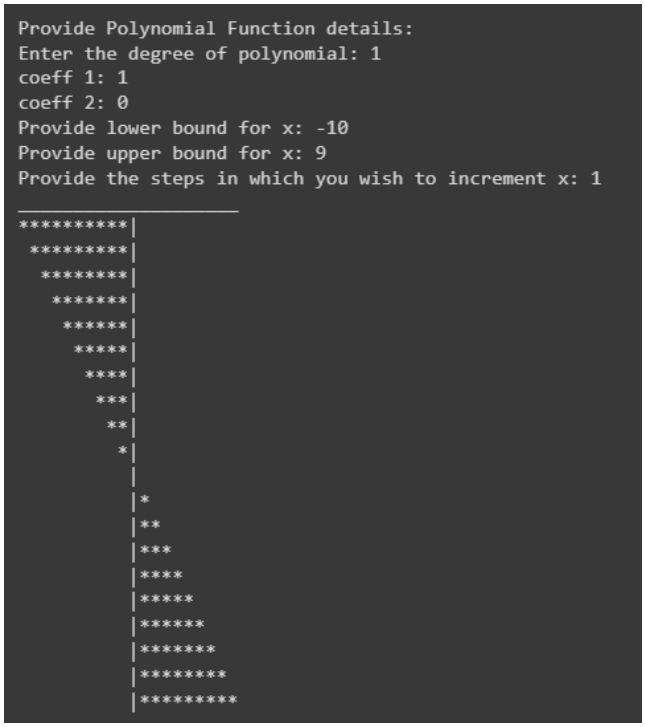
Circle:
Perimeter=43.9822
Area=153.938

Q3 Plot some functions

Yardstick	Marks	Rationale
Code runs on students' system	2	The program should execute at least once without errors.
Explainability	1	Student is able to briefly explain the question and required output.
Use of functions/conditionals	4 (1+1+1+1)	Specific functions/conditional for: ● Degree 0 (1) ● Degree 1 (1) ● Degree 2 (1) ● Degree 3 (1) 2% deduction for the function/conditional not being used, for the kth degree polynomial (k=0 to 3) → no function at all being used, 30 % should be awarded for this component.
Type Handling	2 (0.5+0.5+1)	Handles, in any way: ● Float(0.5) ● Integer(0.5) ● Negative values (1) #step and bounds (lower or upper to be excluded for this consideration)
Runs a sample test case	1	The output should be similar to the provided output (not exact, adding label or axis is

		upto the student, no marking for this)

Sample output (other than the one provided in the assignment)



Some students, who might not have handled the negative values, would be getting the output similar to the lower half of the plot. They should be given full marks for this component as, non-handling of negative values would have already been dealt with in "Type Handling"

Q4.

Yardstick	Marks	Rationale
Correct Implementation	6	1. 3 Nested for loops (one for each boolean variable) (1+1+1) If while loop is used, no issues. Give full marks. Idea is to iterate over the 2 values (True, False) for each variable and check if the expression becomes True. 2. Writing the boolean expression [eg: Fn = b1 and b2] (1) 3. If condition to check if it's satisfiable (1) 4. Printing whether satisfiable or not (if satisfiable, any of the boolean values that make it satisfiable) (1)

Running Testcases	4 (1+1+2)	Both the sample test cases provided in the problem are giving correct output and the test case provided below is giving the correct output
Total	10	

Test Case:

Paste these boolean expressions in the students code and check (as there are no inputs taken in this question)
Example:
If students already have put
`Fn = (b1 and not b1)`
in their code,
replace this line with:
`Fn = (b1 or b2) and (b2 or not b3)`

Sample 1:

```
Fn = (b1 and not b1)
```

Output:
```
Unsatisfiable
```

Sample 2:

```
Fn = (b1 or b2) and (b2 or not b3)
```

Output:
```
Satisfiable
True True True

# Any assignment of the variables can be printed here that makes the
expression satisfiable. These are:
True True True
True True False
True False False
False True True
False True False
```

Test Case:

```
Fn = (b1 or b2) and (b1 or not (b2 and not b3)) and (not b2 and b3)
```

Output:

```
Satisfiable
True False True

# This is the ONLY assignment of the variables that makes Fn true.
```

Q5 :

Points to check	Marks
Menu Is displayed similar to as given in assignment(Some variations is ok). 1) at start or program 2) After processing an operation	0.5 + 0.5
Function **getReverse**(n) is created	0.5
Calling getReverse(12345) prints '54321'	0.3
Calling getReverse(1002001) prints '1002001'	0.3
Function **checkPalindrome**(n) is created	0.5
Calling checkPalindrome(12021) prints True/YES	0.3
Calling checkPalindrome(123421) prints False/NO	0.3
Calling checkPalindrome(89855898) prints True/YES	0.3
Function **checkNarcissistic**(n) is created	0.5
Calling checkNarcissistic(153) prints True/YES	0.5
Calling checkNarcissistic(54748) prints True/YES	0.5
Calling checkNarcissistic(5013) prints False/NO	0.5
Function **findDigitSum**(n) is created	0.5
Calling findDigitSum(26) prints 8	0.5
Calling findDigitSum(1298) prints 22	0.5
Calling findDigitSum(9299) prints 42	0.5
Function **findSquareDigitSum**(n) is created	0.5

Calling findSquareDigitSum(8987) prints 1043	0.5
Calling findSquareDigitSum(87654) prints 441	0.5
Calling findSquareDigitSum(202) prints 8	0.5
After processing an operation, code should not exit but ask user for another operation again. If exit operation is chosen, code must exit.	0.5

Q6.

Category	**Marks**	**Remarks**
Correct Implementation	6	1. Created a list l (0.5) 2. a,b,c, … will be float (0.5) 3. x_0 and threshold will be float (0.5) 4. Created the function find_roots(l) (0.5) 5. Function find_roots(l) returns a root of the function (0.5) 6. Compute f'(x) using loops (0.5) 7. no extra list used while computing f'(x) (0.5) 8. Differentiability check (0.5) 9. Correct newton raphson method implementation inside loop (2)
Running Testcases	4 (1+1+2)	Both the sample test cases provided in the problem are giving correct output and the test case provided below is giving the correct output **Note:** if the output is coming as a complex number but it is closer to the actual answer you can give marks
Total	10	

Test Case -

Sample 1:

c = [2, -1] # so this means $f(x) = x^2 + x^{-1}$

Using Newton-Raphson method starting with $x_0 = 1$ and having a threshold of 0.001

We get the root as -1.0

Sample 2:

c = [2, 1, 1, 1, 0.8] # so this means $f(x) = x^2 + x + x + x + x^{0.8}$

Using Newton-Raphson method starting with $x_0 = 1$ and having a threshold of 0.001

We get the root as 0

Test Case:
c = [3, 5, 5, 123, 51, 22, 4, 22, 41, 22, 5, -2, 8, 7, 215]
Using Newton-Raphson method starting with $x_0 = 1$ and having a threshold of 0.0001
We get the root as -0.849586243711609

Q7.

Category	**Marks**	**Remarks**
Correct Implementation	6	1. Created a list l (0.5) 2. p,q,r, … will be float (0.5) 3. A,b,d, r1,r2 will be float (0.5) 4. Function calculate_area**(l,a,b,d)** returns the area under the curve from r1 to r2 limits as described by the polynomial definition of f(x) using a list of exponents l. (1) 5. Follow correct Simpson's ⅓ algo as provided in the assignment description using loops (3) 6. Checking condition for divisibility of (b-a) by d (0.5)
Running Testcases	4 (1+1+2)	Both the sample test cases provided in the problem are giving correct output and the test case provided below is giving the correct output **Note:** if the output is coming as a complex number but it is closer to the actual answer you can give marks
Total	10	

Example 1:

c = [2, 1] # so this means $f(x) = x^2 + x$
a=0
b=6
d=2
Using Simpson's ⅓ Algo, RHS= 90

Explanation

$$\int_0^6 f(x)\,dx = \int_0^{0+2} f(x)\,dx + \int_{0+2}^{0+2*2} f(x)\,dx + \int_{0+2*2}^{0+2*3} f(x)\,dx = \int_0^2 f(x)\,dx + \int_2^4 f(x)\,dx + \int_4^6 f(x)\,dx$$

$$\int_0^2 f(x)\,dx = \frac{((0+2)-0)}{6}[f(0) + 4f(\frac{(0+(0+2))}{2}) + f(0+2)] = \frac{1}{3}[f(0) + 4f(1) + f(2)]$$

Putting the values, of x = 0, 1, 2 in f(x) we get,

$f(0) = 0,\ f(1) = 2,\ f(2) = 6$

So, $\int_0^2 f(x)\,dx = \frac{14}{3}$

Similarly, do the same step for the other 2 terms,

$\int_2^4 f(x)\,dx = \frac{74}{3}, \int_4^6 f(x)\,dx = \frac{182}{3}$

Finally,

$$\int_0^6 f(x)\,dx = \int_0^2 f(x)\,dx + \int_2^4 f(x)\,dx + \int_4^6 f(x)\,dx = \frac{14}{3} + \frac{74}{3} + \frac{182}{3} = \frac{270}{3} = 90$$

Example 2:

c=[0.5, -1] # so this means $f(x) = x^{0.5} + x^{-1}$

a=1

b=3

d=1

Using Simpson's ⅓ Algo, RHS= 3.897335456369697

Q8.

Category	**Marks**	**Remarks**
Correct Implementation	6	1. Initialized the variables (1) 2. For loop that iterates over 15 years [range(1,16)]. While loop also works. (1) 3. If condition for maintenance calculation under 5 years. (1) 4. Cost calculation, including maintenance and depreciation (1) 5. Utility/Value calculation (1) 6. If condition, when cost becomes less than value i.e. year when we sell the car. (1)
Running Testcases	4 (2+2)	Test cases provided in the problem is giving correct output and the test case provided below is giving the

		correct output
Total	10	

Test Case:

Sample:
Initial cost: 1000000
Depreciation rate: 5% every year
[other initial values given in the question]

Output:
6

Explanation:
You sell it on 6th year
Cost is 475000, Value is 531468.2999

Test Case:
Initial cost: 4000000
Depreciation rate: 5% every year
[other initial values given in the question]

Output:
8

Explanation:
You sell it on 6th year
Cost is 375000, Value is 643076.64299

B1

Yardstick	Marks	Rationale
Code runs on students' system	2	The program should execute at least once without errors.
Explainability	1	Student is able to briefly explain the question and required output.
User inputs + Type handling	2 (1 + 1)	• User input for item prices, discounts, and

		others(1) ● Handles both integer and floats (1)
Function use	3	Atleast 1 function, → 3 Otherwise → 2
Output	1.5	Similar to sample output→ 1 Otherwise 1.5
Formatting	0.5	String formatting properly done, does not look haphazard (0.5)

B2.

Category	**Marks**	**Remarks**
Correct Implementation	6	1. e, d, x_0 , y_0 , z_0, R are inputted correctly (1) 2. e, d, x_0 , y_0 , z_0, R will be float values (0.5) 3. Correct implementation of knowing where a point is wrt sphere (0.5) 4. A loop is used to go to the new point (0.5) 5. No new list/set is made for eg. to store points (1) 6. Keeping Increment a float does not break the code (1) 7. t is maintained <= 1000 8. Correct overall implementation (1)
Running Testcases	4 (2+2)	The sample test cases provided in the problem is giving correct output and the test case provided below is giving the correct output **Note:** if the output is coming a little off due to python float precision you can give marks
Total	10	

Sample Case

For,

e = <0,0,0>

d = <0, 0.8, 0.6>

center = <0.1, 4, 3>
R = 3

We find the first intersection at t=3,
And for the points of intersection, here you can print -
The ray with origin at <0,0,0> and direction <0, 0.8, 0.6> intersects the circle with radius 3 and center at <0.1, 4, 3> first time somewhere between: point <0, 1.6, 1.2> at t=2 and point <0, 2.4, 1.8> at t=3

We find the second intersection at t=8,
And for the points of intersection, here you can print -
The ray with origin at <0,0,0> and direction <0, 0.8, 0.6> intersects the circle with radius 3 and center at <0.1, 4, 3> second time somewhere between: point <0, 5.6, 4.2> at t=7 and point <0, 6.4, 4.8> at t=8

Test Case
For,
e = <6.7,4.7,3.6>
d = <13,14.8,7.6>
center = <13,14.8,7.6>
R = 15

We find the first intersection at t=2,
And for the points of intersection, here you can print -
The ray with origin at <6.7,4.7,3.6> and direction<13,14.8,7.6> intersects the circle with radius 15 and center at <13,14.8,7.6> first time somewhere between: point <19.7, 19.5, 11.2> at t=1 and point <32.7, 34.3, 18.8> at t=2