CSE101: Assignment 1

1: Printing Patterns

Approach

For each question, the number of spaces to be printed in each line have to be calculated by finding out the line number, `i` and multiplying `" "` by `n-i-1`

2: Geometry

References

- 2D Shapes: Perimeter and Surface Area Formulas, ThoughtCo.
- 3D Shapes: Area and Volume of Combination of Solids, toppr

3: Plotting Polynomials

Approach

The value of a polynomial function at each value of `x` can be calculated by taking a variable `sum = 0`, and then for each term in the polynomial, taking `x` to the power of the `degree` of that term, multiplying it by the `coefficient` and adding it to `sum`.

Negative values can be handled by first calculating the most negative value in the range, printing the same number of spaces before the `|` for `x` where `f(x) > 0` and by printing `minimum value - f(x)` spaces before the `*`s for `x` where `f(x) < 0`.

4: Satisfiability Problem

Approach

Since we're given that the variables are `b1`, `b2` and `b3`, we can simply loop over the three of them for any given boolean function

While A1_2021066_4.py executes the statements given in the problem statement, I have made A1_2021066_4alt.py which can input any such boolean function and check for its satisfiability by using the `eval()` function.

5: Numbers

Approach

For `findDigitSum()` and `findSquareDigitSum()`, recursion has been sued to iterate till a single digit number has been achieved.

References

- Narcissistic Number

7: Simpson's 1/3 Algorithm

Approach

Simpson's 1/3 Algorithm has been followed to approximate the area. An `if-else` condition is used to check if `b-a` is divisible by `d`

A for loop is run from `a` to `b` with step `d` and the area for each step is calculated as `d/6[f(i) + 4f(i+d/2) + f(i+d]`)

The loop is terminated before `b` instead of after `b` because we want the area till `b`, not till `b+d`

The value of a polynomial function at each value of `x` can be calculated by taking a variable `sum = 0`, and then for each term in the polynomial, taking `x` to the power of the `degree` of that term and adding it to `sum`.

References

Simpson's Formula

Bonus 2: Ray Sphere Intersection

Approach

The distances from the starting point of the ray to the intersection points, `t1` and `t2` can be calculated by solving a quadratic equation of the form `ax^2 + bx + c` where `a = 1, b = 2 * dot(e - c, d) and c = |e|^2 - r^2`.

Note: `dot()` represents dot product of two vectors.

Finally, the vector equation `e + td` for `t1` and `t2` gives the intersection points. However, this condition assumes a line intersecting a sphere, and not a ray. Hence, we only pick the positive `t(s)`.

References

- Raytracing - Ray Sphere Intersection: 1000 Forms of Bunnies
- Dot product and cross product of two vectors: GeeksForGeeks