Avery Nelson --- 0 late days used

010920903

[amn018@uark.edu](mailto:amn018@uark.edu)

**Artificial Intelligence Assignment 1 Report**

**Description of Implementation**

The goal of this assignment is to implement a base class and the constructors for 5 classes. This required that the 5 classes inherit the base class, one of which inherits it through another one of the 5 classes. This required creating a function meant to count the number of instances of the base class, then implementing constructors and area functions of each of the 5 shape classes. This was fairly straightforward and mostly just required implementing code to calculate the basic area functions for a triangle, rectangle, square, circle, and polygon respectively.

The next requirement was to implement funct6ions for matrix calculations. The first step was meant to multiply the two given matrices without using np.dot, .dot, or & operator of NumPy library in the implementation. I ran some simple checks then used nested loops. While not particularly efficient, this program required relatively little out of the nested loops and as such the time was not a significant issue. Then the assignment required that a recursive function and an iterative function both be used to perform the powers of a matrix. Neither was particularly hard to implement or test, and differed in approach, not the outcome, proving that they worked correctly.

A math equations and symbols

Description automatically generated with medium confidenceThe next task of the assignment was calculating the *nth* Fibonacci number using powers of the matrix *A*. This step had built in testing, so it was verified easily. It required calling the recursive power function of the matrix A and was confirmed by the assertions in the testing functions. Then, we needed to implement a function that calculates the *fn* of a recursive function below

using power of matrix. I had more issues with this problem because the testing did not announce if the results were valid, so upon checking I realized that my initial function was not correct. My correct implementation required cycling through the range and appending results (up to the required number) then summing the requested number of preceding values.

The final requirement was to implement a depth-first search and a breadth-first search based on a provided *mxn* matrix with a zero-based index. With a value of 0, a cell cannot be accessed. The goal is to find a path that reaches the bottom right corner of the matrix. Both of these are standard algorithms that require tracking the cells that were visited and determining which valid cells are available to move to next while attempting to reach the goal node. The final goal (and advanced problem) was to find the minimum cost path through the matrix, much the same way the last problem was solved, but with the goal of minimizing the total cost of the path. This is solved by determining the cost of valid paths and choosing the one with the minimum cost.

**Results**

Overall, my implementations of each of the problems appears to be successful. Areas I could confirm manually, which the tests of both matrix multiplication and the Fibonacci confirmed their accuracy for me. I ran a test case of the recursive and iterative powers because the randomized number made it hard to confirm that both were returning the same result and thus working in the same way. I also confirmed the f(n, k) outputs and searches manually since confirmation was not built-in. f(n,k) aligns with what the manually calculated sum of the selected previous numbers should be. The results of the DFS and BFS implementations both also give valid paths spanning from the start node to the goal node. The minimum then calculates the path with the lowest cost of the available valid paths. Thus, the project successfully completes the required tasks given the test cases and parameters provided.

A screenshot of a computer program

Description automatically generated

A computer screen shot of a blue screen

Description automatically generated

A computer screen shot of a blue screen

Description automatically generated