Spike: 08
Title: Goal-Oriented Action Planning (GOAP)

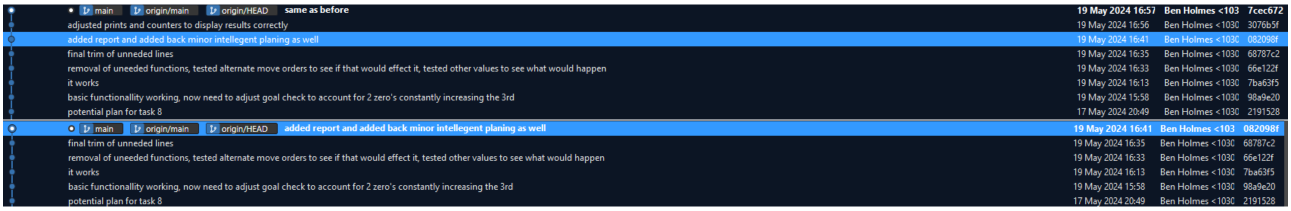
Author: Ben Holmes, 103024841

Goals / deliverables:
1. Create a GOAP simulation that demonstrates the effectiveness of the technique in considering long-term outcomes of actions (related to side-effects and/or time delays) and can plan and act intelligently.

Technologies, Tools, and Resources used:

- Visual Studio Code
- Python 3.12.2

Tasks undertaken:



```
main  origin/main  origin/HEAD  same as before                                                       19 May 2024 16:57  Ben Holmes <103   7cec672
adjusted prints and counters to display results correctly                                            19 May 2024 16:56  Ben Holmes <1030  3076b5f
added report and added back minor intellegent planing as well                                        19 May 2024 16:41  Ben Holmes <1030  082098f
final trim of unneded lines                                                                          19 May 2024 16:35  Ben Holmes <1030  68787c2
removal of uneeded functions, tested alternate move orders to see if that would effect it, tested other values to see what would happen   19 May 2024 16:33  Ben Holmes <1030  66e122f
it works                                                                                             19 May 2024 16:13  Ben Holmes <1030  7ba63f5
basic functionallity working, now need to adjust goal check to account for 2 zero's constantly increasing the 3rd   19 May 2024 15:58  Ben Holmes <1030  98a9e20
potential plan for task 8                                                                            17 May 2024 20:49  Ben Holmes <1030  2191528
```

```
main  origin/main  origin/HEAD  added report and added back minor intellegent planing as well        19 May 2024 16:41  Ben Holmes <103   082098f
final trim of unneded lines                                                                          19 May 2024 16:35  Ben Holmes <1030  68787c2
removal of uneeded functions, tested alternate move orders to see if that would effect it, tested other values to see what would happen   19 May 2024 16:33  Ben Holmes <1030  66e122f
it works                                                                                             19 May 2024 16:13  Ben Holmes <1030  7ba63f5
basic functionallity working, now need to adjust goal check to account for 2 zero's constantly increasing the 3rd   19 May 2024 15:58  Ben Holmes <1030  98a9e20
potential plan for task 8                                                                            17 May 2024 20:49  Ben Holmes <1030  2191528
```

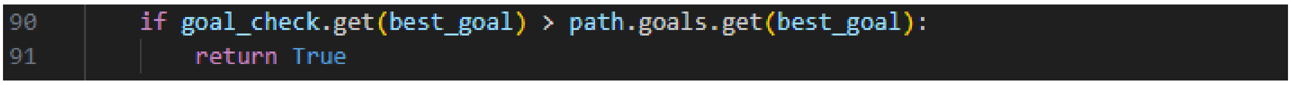
- Copied gop from task 7
- Adjusted to use 3 values instead of 2 and renamed them to energy, hunger and fitness (fitness still wants to go to 0, its lack of fitness shortened to fitness for ease of typing)
- Created new recursive function for path planning called action_paths, it loops through each possible action and applies it to a temp instance of the goals, in the loop this temp instance of the goals is then passed to another action_paths which loops through the next moves (move number 2). This eventually generates a path (moves done and final goals of moves done) in a Path object that is then added to a list of path objects called paths which is returned. Action paths also has a temp_paths which is looped through and appended to paths if another action_paths is called
- Action_paths is called in the chose_action_path function, which if a path has not been generated before, generates a paths list and then loops through it checking against the first paths goals to see if the path produces better results. (it also compares length of path and choses the shortest path if one is shorter as there is a check in action_paths that shortens the path if 0 0 0 is hit) The best path is stored in the global variable, the path moves are then iterated through till the move_counter (number of path moves) hits the move_amount (max number of path moves), once move_amount is hit, the counter is reset and next time chose action_paths is called, then the paths are regenerated.

What we found out:

For the deliverable, the simulation was created to have 6 actions with different increases and decreases and stability of the 3 goals so that the planning function had to account for side effects of its actions. The intelligent actions are determined in 2 ways, first is checking to see if the move count is less than the move depth (or current best path) then it choses that path. The next intelligent check is when it checks for the decrease in the largest goal after checking for all 3 values decreasing. This is to make each path more efficient. The second check improves efficiency at depth 3-4 but depth 5 its nearly identical.

If you wish to test the depth side of things, simply change the move depth number, although be warned at depth 6 and above it becomes much slower each increment.

If you wish to test the efficiency of the largest goal check please comment out these lines



```
    if goal_check.get(best_goal) > path.goals.get(best_goal):
        return True
```