ממ"ן 11

מוטי עזרן - 318298734

שאלה 1

בשאלה זו כתבתי פונקציית חיפוש כללית שמקבלת מבנה נתונים שייצג את ה"גבול", במקרה של DFS מבנה הנתונים יהיה מסוג תור (Queue).
אלגוריתם DFS אינו מוצא את הפתרון האופטימלי מפני שהוא מחפש לעומק ומחזיר את המסלול הראשון שהוא פוגש ומסלול זה לא בהכרח יהיה הכי קצר, ולכן לא בהכרח שהפתרון שהחיפוש ימצא יהיה הפתרון הזול ביותר.

שאלה 2

בשאלה זו השתמשתי בפונקציית החיפוש הכללית שכתבתי והמבנה נתונים שמייצג את ה"גבול" בשאלה זו הוא מסוג מחסנית (Stack).
אלגוריתם BFS מוצא את הפתרון האופטימלי כאשר הגרף שהוא מחפש בו לא ממושקל, ובשאלה זו לכל צעד יש אותו מחיר ולכן BFS בהכרח מוצא את הפתרון הזול ביותר..

שאלה 3

בשאלה זו גם השתמשתי בפונקציית החיפוש הכללית שכתבתי והמבנה נתונים שמייצג את ה"גבול" בשאלה זו הוא תור עם עדיפות, כלומר תור שיכול לקדם אובייקטים מסוימים לפי המחיר שלהם, ולכן האובייקט עם המחיר הנמוך ביותר יצא ראשון מהתור ובמקרה של UCS העדיפות של כל אובייקט היא סכום המחירים של הפעולות במסלול.

שאלה 4

בשאלה זו גם השתמשתי בפונקציית החיפוש הכללית שכתבתי והמבנה נתונים שמייצג את ה"גבול" בשאלה זו הוא תור עם עדיפות, כאשר העדיפות של כל אובייקט היא הסכום של מחיר הפעולה והיורסטיקה של הפעולה והעדיפות של כל פעולה הוא סכום העדיפויות של כל האובייקטים במסלול.
כשאנו מריצים את הפקמן ב־openMaze אנו מקבלים ב־DFS מסלול יקר יחסית במחיר של 298 כשהוא פיתח 576 מצבים.

כשאנו מריצים את הפקמן ב־openMaze בעזרת חיפוש BFS או UCS אנו מקבלים אותה תוצאה (כי אין משקלים לפעולות) מסלול זול במחיר של 54 כשהם מפתחים 682 מצבים.
כשאנו מריצים את הפקמן ב־openMaze בעזרת חיפוש AStar אנו מקבלים אותה תוצאה כמו BFS כשמשתמשים ב־nullHeuristic אבל כשמשתמשים ב־manhattenHeuristic מקבלים מסלול זול במחיר של 54, כמו ב־BFS, אבל מפתחים רק 535 מצבים.

שאלה 5

הוספתי למצב ההתחלתי רשימת פינות שמייצגת את הפינות שעוד לא עברנו במסלול, ולכל מצב שמפותח (Successor) נבדק האם הוא פינה ואם כן מורידים אותו מרשימת הפינות (כי עברנו אותו במסלול). לבסוף כשהרשימה מתרוקנת אנחנו אומרים שהגענו למצב המטרה.

שאלה 6

היוריסטיקה שכתבתי בשאלה זו סוכמת את המרחק לפינה הקרובה ביותר וממנה לפינה הקרובה אליה וכן הלאה לכל הפינות. היורסטיקה תמיד מחשבת את מרחק מנהטן לפינה הקרובה ביותר וכל מרחק כזה הוא קטן או שווה למרחק האמיתי שפקמן יעשה, כלומר $h(n) \leq h^*(n)$ ולכן היוריסטיקה קבילה. היוריסטיקה היא גם עקבית כי אם פינה x נבחרת כפינה הקרובה ביותר למיקום אז מרחק שאר הפינות ממנה הוא קבוע (כי מיקום הפינות קבוע) ולכן ככל שאנו מתקרבים לפינה x היוריסטיקה תקטן ולכן היוריסטיקה היא גם עקבית.

שאלה 7

היוריסטיקה שכתבתי בשאלה זו מחזירה את המרחק האמיתי (כמות הצעדים החוקיים שצריך לעשות כדי להגיע לאוכל) אל האוכל הרחוק ביותר. יוריסטיקה זו היא קבילה כי במקרה הטוב כל האוכל יהיה בין המיקום הנוכחי לאוכל הרחוק ביותר, וזהו תרחיש אופטימי ולכן ברור שמתקיים $h(n) \leq h^*(n)$ ולכן היוריסטיקה קבילה. וברור שהיא גם עקבית כי עם כל צעד לכיוון האוכל הרחוק ביותר ערך הפונקציה יקטן באחד לכל היותר. תחילה חישבתי כל פעם את המרחק מהמיקום לאוכל אך פעולה זו לקחה הרבה זמן כי היו מרחקים שחישבתי מספר פעמים ולכן שמרתי במילון heuristicInfo לכל מיקום ואוכל את המרחק שלהם וכך חסכתי את חישוב המרחק בפעמים הבאות בהם נתקלתי בו וייעלתי את זמני הריצה של הפונקציה.

שאלה 8

בשאלה זו הלכתי לפי הרמז וקודם כל השלמתי את החסר ב־AnyFoodSearchProblem בבעיה זו אנו פשוט רוצים להגיע לאוכל, לא משנה מאיזה סוג, ולכן מצב המטרה של בעיה זו הוא כל מצב שבו יש אוכל אז פשוט החזרתי מהפונקציה

isGoalState של בעיה זו בוליאני המתאר האם יש אוכל במצב. לאחר שפתרתי בעיה זו עשיתי חיפוש AStar על הבעיה AnyFoodSearchProblem והחזרתי את המסלול שהתקבל.