توضیحات بخش dfs

در ابتدا یک استک میسازیم، این استک مجموعه ای از مسیر (لیست) تا رسیدن به یک استیت را ذخیره میکند، این مسیر شامل سه تایی متشکل از استیت، اکشن مورد نظر برای رسیدن به آن استیت و هزینه رسیدن به آن استیت را دارد

به طور خلاصه لیستی از این سه تایی ها یک مسیر را تشکیل میدهد، و استک مجموعه ای از این مسیر هاست

سپس استیت شروع مسئله را بدست آورده، و لیستی شامل سه تایی مربوط به این استیت به استک اضافه میکنیم.
همینطور یک لیست visited شامل استیت شروع داریم، که با استفاده از آن از بررسی استیت های تکراری جلوگیری میکنیم، در نهایت جست و جو را آغاز میکنیم

تا زمانی که استک خالی نشده، یک مسیر از آن خارج میکنیم، آخرین سه تایی مسیر را به دست آورده، و استیت آن را نیز به دست می آوریم

اگر استیت مورد نظر هدف مسئله بود، یک لیستی از اکشن های انجام شده در طور مسیر تا رسیدن به آن استیت را به عنوان جواب برمیگردانیم

در غیر این صورت، لیست سه تایی های همسایه های استیت را به دست آورده، اگر استیت همسایه داخل visited موجود نبود، آنرا اضافه میکنیم، سپس یک مسیر جدید شامل مسیر قبلی به علاوه سه تایی همسایه جدید درست کرده و به استک اضافه میکنیم

توضیحات *A

برای این بخش ابتدا یک کلاس نود تعریف کرده که اطلاعات داخل آن شامل سطر و ستون خانه مورد نظر و مقادیر g، h و f نود مورد نظر، مسیر رسیدن به نود مورد نظر، استیت مربوط به آن نود و اکشن مورد نیاز برای رسیدن به آن نود است، همینطور اپراتور های less than و greater than بر اساس مقدار f تعریف شده، و اپراتور eq بر اساس برابری استیت ذخیره شده در کلاس نیز تعریف شده است، سازنده این کلاس مسیر والد را کپی کرده و سپس اکشن را به آن اضافه میکند

همینطور یک تابع برای به دست آوردن فاصله منهتنی بین دو نود از کلاس ذکر شده تعریف شده است

در تابع اصلی سرچ ابتدا نود هدف را تعریف کرده، که سطر آخر و ستون اول است، سپس نود شروع را که سطر اخر و ستون اول است با استیت اولیه مسئله میسازیم

همینطور یک لیست open و یک لیست closed برای استفاده در الگوریتم تعریف کرده و نود شروع را در آن قرار میدهیم، سپس جست و جو را آغاز میکنیم

تا زمانی که openlist خالی نشده، آن را سورت کن (به دلیل ساختار توابع lt و gt در کلاس نود، لیست بر اساس f سورت میشود) سپس اولین نود آن را از لیست خارج کن، و به لیست closed اضافه کن

اگر نود مورد نظر هدف مسئله بود، مسیر رسیدن به آن نود را به دست آورده و به عنوان جواب باز میگرداند

در غیر این صورت، فرزندان نود مورد نظر را بدست آورده، اگر فرزندی داخل closedlist بود آن را نادیده میگیرد، در غیر این صورت، سطر و ستون فرزند را با توجه به اکشن انجام شده برای رسیدن به آن به دست آورده، نود مربوط به آن فرزند را ساخته، مقدار g را بر اساس g مادر و هزینه رسیدن به آن نود، مقدار h را بر اساس تابع فاصله منهتن و مقدارf را بر اساس g و h به دست میاورد، سپس اگر این نود در لیست باز قرار داشت و مقدار g آن بیشتر از نود داخل لیست بود آن را نادیده میگیرد، در غیر این صورت فرزند را به لیست باز اضافه میکند