3-c 19257514 奈良 貴明　計算工学基礎レポート

メールに解説付きソースコードを
code.zip{cs3b-LongestPath.cpp,list2.cpp,list2.hpp,cs3-c.cpp,result-longestPath.txt,result-cs3c.txt}として添付した.

・実行結果

　cs3b-LongestPath.cppをコンパイルし実行した結果を，計算結果が多く表示量が多くなるためリダイレクトを用いてファイル化しresult-longestPath.txtとして添付した.
実行結果から，全パスを探索し最長経路でゴールまで行く経路を表示できていることが確認できる.

　cs3-c.cppをコンパイルし実行した結果を，同様の理由でリダイレクトを用いてファイル化しresult-cs3c.txtとして添付した.
実行結果から，すべての出発点から幅優先探索を行い，最短の到達点を見つけ最短経路で結んでいることが確認できる.

・考察，感想（最長経路）

　自分が考えた最長経路探索のアルゴリズムを説明する.

　はじめに，注目点（スタート座標）の四方について移動可能かをチェックし，移動可能な場合その座標をスタックにpushしていく．四方をチェックし終えたらスタックから次の注目点をpopする.
このようにして迷路を進んでいくと分岐がない直線経路の座標がスタック上から消えていき分岐先の座標だけがスタック上に残されていく．もし，注目点が移動不可能になった場合，スタックから次の移動座標をpopすると一番最後に分岐あった座標の別の分岐先の座標が注目点になる．このとき，それまで探索していたパスは消去する.
これをスタックが空になるまで繰り返すことで全パスを列挙することができる.

　注目点がゴールにたどり着いた場合，ゴールまでのステップ数がそれまで見つかっているゴールまでのステップ数より大きければ結果を更新する.

今回のプログラムは，アルゴリズムを思いつくまでに時間がかかりかなり雑な実装になってしまった．また全パスを探索しているため，実行時間が長く非効率である．高速化のアイデアとして，注目点から移動可能点を探すのを隣接するマスだけではなく2マス先，3マス先まで一回で探索すれば，スタック上にpushする座標数を減らすことができるのではないかと考えた.

・考察，感想（複数出発点，複数到達点）

　複数出発点，複数到達点のアルゴリズムを説明する.

　はじめに，迷路から出発点を探しすべてキューにputする．こうすることによってすべて出発点から幅優先探索を始めることができる．あとは，幅優先探索を進めていき，最初にゴールにたどり着いた経路を最短経路としそこからステップ数を辿って最短出発点を結び表示する.

　この課題は，出発点を複数扱えるようにするだけで良かったので非常に簡単だった.

・協力者
19257504 太田 那菜