

Задача ДОСТИЖИМОСТ

Пояснение към решенията

Решението за 54 т. – файл reach_54p.cpp

Данните от таблицата се прочитат в масива `int a[M][N]`. В проходимите клетки се записва числото `DOT=-1`, а в непроходимите – числото `AST=-2`. Допълнително таблицата се огражда с контур, състоящ се от ред 0, ред $m + 1$, колона 0 и колона $n + 1$, в които се записва `AST=-2`, за да се избегне проверката за излизане извън таблицата.

Прилага се вариант на алгоритъм за търсене в ширина, известен като “Вълна”. За целта във функцията `int wave(int is, int js)` първо се маркира стартовата клетка с 0 и след това в цикъла `while(b)` многократно се обхождат всички клетки на таблицата. При всяко обхождане в клетките се записва поредната стойност на `d`, което дава отдалечеността (нивото) на клетката от стартовата. За целта от всяка клетка от текущото ниво се търсят кои клетки са нейни съседи и в тях се записва `d+1`. Така `d` за клетката показва с колко най-малко стъпки може да се достигне тази клетка. Процесът продължава, докато е възможно намиране на съседи. При прекратяването на процеса, стойността на `d` показва на колко стъпки е най-отдалечената клетка от стартовата.

За всяка поредна двойка координати (i, j) , за които трябва да се пресметнат стъпките до най-отдалечената достижима клетка се извиква `int d=wave(i, j)`, което пресмята това. Поредната пресметна стойност се извежда в изхода на програмата. След всяко използване на функцията `wave` се извиква функцията `clear()`, която възстановява началното състояние на масива `a[][]`.

Решението за 100 т. – файл reach_100p.cpp

Функцията `int wave(int is, int js)` от предишната програма тук заменяме с по-бързо работеща функция `int bfs(int i, int j)` за търсене в дълбочина. Използва се опашка `queue<pair<int, int>>q` от STL. В тази опашка първоначално се вкарва двойката координати на стартовата клетка. След това в цикъла `while(!q.empty())` се изкарва по една двойка координати `auto p = q.front(); q.pop(); i=p.first; j=p.second;` на клетка и за тази клетка се търсят всички нейни съседи с координати `i0, j0`, които не са били посетени. За тези клетки се записва ниво с единица по-голямо от предишното и след това тези клетки се вкарват в опашката. В променливата `d` се поддържа най-голямото намерено ниво, което след завършване на цикъла се подава като стойност на функцията `bfs`.

Алтернативно решението за 100 т. – файл reach_alt_100p.cpp

Задачата може да се реши без използване на опашка, а като модифицираме функцията `wave` от `reach_54p.cpp`. Няма да правим обхождане на всички клетки (i, j) от таблицата чрез двоен цикъл за всички индекси (i, j) , а ще използваме само тези индекси, които са на клетките от предишното ниво. За целта използваме два вектора `pi` и `pj` за записване на индексите от предишното ниво и още два вектора `pni` и `pnj`, в които ще записваме новите индекси на клетките от текущото ниво. В края на обработката на нивото записваме индексите от `pni` и `pnj` съответно в `pi` и `pj`.

Емил Келеведжиев