자료구조 HW4

B935394 컴퓨터공학과 장준희

October 26, 2020

1 stack의 작동방식

stack은 이름에서부터 알 수 있듯이, 하나하나 쌓여진 자료구조이다. 그렇기 때문에 나중에 들어온 것이 가장 먼저 나가는 후입선출을 하게된다. 다양한 타입에도 stack 자료구조를 사용할 수 있게끔 template로 작성되어 있다. push(),pop(),empty(),top()등의 함수를 가지고 있는데 각각의 역할은 다음과 같다.

- push():스택에 주어진 데이터를 얹는다.
- pop():주어진 스택의 맨 위 데이터를 삭제한다.
- empty():스택이 비어있는지 여부를 알려준다.
- top():스택 맨 위의 데이터를 반환한다.

2 경로를 찾는 알고리즘과 stack의 활용

미로를 좌표에 올려서 생각을 시작한다. 현재 좌표에서 북,북동,동,남동,남,남서,서,북서의 여덟방향으로 한 칸 씩 움직일 수 있다. 미로배열과 각 좌표를 다녀간적 있는지 여부를 파악하기 위한 mark배열, 그리고 이동한 내역을 차곡차곡 담을 **stack**이 필요하다. 처음의 이동만 입력을 해주면, 나머지는 반복문에서 해결할 수 있다. 처음의 이동을 stack에 넣어준 후, 반복문1에서 꺼내 읽는다. 다음 지점이 새로운 장소라면, 현재장소와 (방향+1)을 스택에 저장한다.(다음 지점에서 그 다음으로 이동할 수 없다면, 반복문2를 나와서 저장한 장소와 방향을 읽어야(unstack) 해야한다.) (우리가 경로를 읽을 때는 실제 이동방향은 필요없고 좌표만 필요하므로 방향+1이 stack되어도 상관없다.)그렇게 stack과 unstack을 반복하면서, 경로를 저장한다.

```
void Path(const int m, const int p){
    mark[1][1]=1;   // start at (1,1)
    stack<Items> pathStack;
    Items tempItem(1,1,E);
    pathStack.push(tempItem);
    int numOfVisitedNode=0;
    while(pathStack.empty()!=true){         //stack is not empty,반복문1
        tempItem=pathStack.top();  pathStack.pop(); //unstack,정보얻기or되돌아가기
        int i=tempItem.x;   int j=tempItem.y;   int d=tempItem.dir;
        while(d<8){         //(i,j)에서(g,h)이동,반복문2
            int g = i+mov[d].a; int h = j + mov[d].b;   //the point where we try to go
            if((g==m)&&(h==p)){         //reached exit
                cout<<pathStack;        //  print path
                tempItem.x = i; tempItem.y = j; cout << " -> " << tempItem;
                        //print present point::cout<<" -> ("<<i<<","<<j<<")"도 가능..
                tempItem.x = m; tempItem.y = p; cout << " -> " << tempItem << endl;
                        //print next point
                cout<<"\n#node visited = "<<numOfVisitedNode+1<<" out of "<<m*p<<endl;
                        //아직 도착지는 방문 안했기때문에 +1만.
                return;
            }
            if((maze[g][h]==0)&&(mark[g][h]==0)){
                //next point is new, possible point **!something<==>something==0, 착각하지말자
```

```
            mark[g][h]=1;   numOfVisitedNode++;
            tempItem.x=i;tempItem.y=j;tempItem.dir=d+1;
                    //present point, if failed, next trial direction
            pathStack.push(tempItem);//stack한다
            i=g;j=h;d=N;    //N(=0)방향부터 시계방향으로 시도해본다.
        }       //not understand yet
        else{
            d++;
        }
    }
    //8방향을 다 둘러보았지만, 갈 수 있는 곳이 없음
}
cout<<"No path in maze"<<endl;
```

3 힘들었던 점

- path()에서 unstack을 하는 이유와 반복문2 안의 두번째 if문의 과정이 이해가 안갔었다. 적당한 예시를 들고 코드를 한줄한줄 손으로 그림을 그려서 표현해보니 왜 unstack을 하는지 if문 안의 d+1을 하는 이유 등을 알 수 있었다.
- main함수의 형식이 그간 봐오던 것이 아닌 main(int argc, char* argv[])여서 당황스러웠다. 구글링을 통해서 파일 입출력에 쓰이는 형식임을 알 수 있었다.
- offset이 무엇인지 몰랐다. c++공부를 항상 해야겠다고 생각했다.