```pascal
program MazeGame;
uses
  sysUtils;
type

    RoomPtr = ^Room;

    Direction = (North, South, East, West);

    maze = Array of RoomPtr;

    Door = Record
      Heading     : Direction;
      Destination : RoomPtr;
    end;

    Room = Record
        Title       : String;
        Description : String;
        IsGoal      : Boolean;
        doors       : Array of Door;
    end;


function CreateRoom(title, description : String; goal:Boolean):RoomPtr;
var
  newRoom: RoomPtr;
begin
  // this allocates space on the HEAP
  New(newRoom);

  // put data into newRoom
  newRoom^.Title := title;
  newRoom^.Description := description;
  newRoom^.IsGoal := goal;
  SetLength(newRoom^.doors, 0);
  // return new room
  result := newRoom;
end;

function DirToString(dir: Direction): String;
begin
  case dir of
  North: result := 'North';
  South: result := 'South';
  East: result := 'East';
  West: result := 'West';
  else result := 'Unknown room';
    end;
end;

procedure DisplayRoom(toShow: RoomPtr);
    var
    i : Integer;
    begin
      WriteLn('------------');
      WriteLn(toShow^.Title);
      WriteLn('------------');
      WriteLn(toShow^.Description);
      WriteLn('-------------------------------------------------');
      for i := Low(toShow^.doors) to  High(toShow^.doors)do
        WriteLn(' ', i+ 1,': ', toShow^.doors[i].heading) // this is an enumerated t
end;

procedure AddDoor( var fromRoom: RoomPtr; heading: Direction; toRoom: RoomPtr);
begin
  SetLength(fromRoom^.doors, Length(fromRoom^.doors)+1 );
  fromRoom^.doors[High(fromRoom^.doors)].heading := heading;
  fromRoom^.doors[High(fromRoom^.doors)].destination := toRoom;
end;

procedure LoadMaze(filename: String; var myMaze: maze; var player : RoomPtr);
var
  input: Text;
  space: Char;
  dir: Direction;
  title, desc: String;
  roomCount, exitCount, goalIdx: Integer;
  i, fromRoom, toRoom: Integer;

begin
  Assign(input, filename);
  Reset(input);

  ReadLn(input, roomCount);
// set the length of the array myMaze
  SetLength(myMaze,roomCount);
  ReadLn(input, goalIdx);
  for i := 0 to roomCount - 1 do
  begin
    ReadLn(input, title);
    ReadLn(input, desc);
    if (i <> goalIdx) then
    begin
      myMaze[i] := CreateRoom(title, desc, false);
    end
    else
    begin
     myMaze[i] := CreateRoom(title, desc, true);
    end;
  end;

  ReadLn(input, exitCount);
  for i := 0 to 19 do
    begin
    ReadLn(input, fromRoom, space, toRoom, space, dir);
    AddDoor(myMaze[fromRoom-1], dir , myMaze[toRoom-1]);
    end;

  Close(input);
end;

procedure Main();
var
  myMaze : maze;
  player : RoomPtr;
  option : Integer;
begin
  // Write the Loadmaze procedure
  LoadMaze('maze.txt', myMaze, player);

    WriteLn('------------------------------------------------');
  player := myMaze[0];
 repeat
```

```
  DisplayRoom(player);
  Write('Take exit: ');
  ReadLn(option);
  player := player^.doors[option- 1].destination;
 until player^.IsGoal;
    WriteLn('-------------------------------------------------');
    DisplayRoom(player);
end;

begin
  //DemoFileReading('Maze.txt');
  Main();
end.
```