Exercise 5

Task 1

```
payment(0, []). % base case for recursion
payment(Amount, [Head|Tail]) :-
    % extract info from head
    Head = coin(AmountNeeded, ValOfCoin, AmountAvail),
    % constrain AmountNeeded (for search)
    AmountNeeded in 0..AmountAvail,
    % find the next amount to find by subtracting from
    % the current amount
    NextAmount #= Amount - AmountNeeded * ValOfCoin,
    % recursive call
    payment(NextAmount, Tail).
```

Task 2

Task 2.1

```
% base case for recursion
path(Cabin1, Cabin2, Path, TotalDistance, _) :-
    distance(Cabin1, Cabin2, TotalDistance, 1),
    Path = [Cabin1, Cabin2].

path(Cabin1, Cabin2, Path, TotalDistance, Seen) :-
    % check if we have reached the end
    not(Cabin1 = Cabin2),
    % find a path between two cabins
    distance(Cabin1, X, Distance, 1),
    % check if we already have visted this cabin
    not(member(X, Seen)),
    % append X to seen list
    append(Seen, [X], NewSeen),
    % append Cabin1 to the SubPath
    append([Cabin1], SubPath, Path),
    % update totaldistance
    TotalDistance #= Distance + SubDistance,
    % recursive call
    path(X, Cabin2, SubPath, SubDistance, NewSeen).

% main predicate
plan(Cabin1, Cabin2, Path, TotalDistance) :-
    % call to recursive function
    path(Cabin1, Cabin2, Path, TotalDistance, [Cabin1]).
```

Task 2.2

Spagetthi and does not work.

```
findShortestPath([], _, _).
findShortestPath([PathsHead|PathsTail], ShortestPath, ShortestDistance) :-
    PathsHead = [Path, TotalDistance],
    (   TotalDistance < ShortestDistance ->
        ShortestPath = Path,
        ShortestDistance = TotalDistance
    ;   true
    ),
    findShortestPath(PathsTail, ShortestPath, ShortestDistance).

shortestPath([PathsHead|PathsTail], ShortestPath, ShortestDistance) :-
    PathsHead = [Path, TotalDistance],
    ShortestPath = Path,
    ShortestDistance = TotalDistance,
    findShortestPath(PathsTail, ShortestPath, ShortestDistance).

bestplan(Cabin1, Cabin2, Path, Distance) :-
    bagof([Path, TotalDistance],
          plan(Cabin1, Cabin2, _, TotalDistance),
          Paths
    ),
    shortestPath(Paths, Path, Distance).
```