CS 3520: Algorithms

Homework 7

**Due Date: Wednesday, May 1, 2019 at beginning of class**

***Please write legibly. Show your steps to receive partial credit.***

1. (50 points) Solve the coin row problem if the coins are 3, 5, 2, 6, 7, 4, 9. The goal is to pick up the maximum amount of money subject to the constraint that no two coins adjacent in the initial row can be picked up. Find the maximum amount that can be picked up and which coins should be picked up. Show your steps to get the solution.

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Index | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
| Coins | - | 3 | **5** | 2 | 6 | **7** | 4 | **9** |
| F(n) | 0 | 3 | 5 | 5 | 11 | 12 | 15 | 21 |

For maximum you grab: 5, 7, 9 or 2, 3, 7, 9

F(1) = 3

F(2) = max(3, 5 + 0) = 5

F(3) = max(5, 2+3) = 5

F(4) = max(5, 6+5) = 11

F(5) = max(11, 7 + 5) = 12

F(6) = max(12, 11 + 4) = 15

F(7) = max(15, 12 + 9) = 21

1. (50 points) Several coins of different values are placed in cells the following board. The numbers outsides the boxes are row and column numbers. The numbers inside the boxes are coin values. A robot, located in the upper left cell of the board, needs to collect as much money from the coins as possible and bring them to the bottom right cell. On each step, the robot can move either one cell to the right or one cell down from its current location. When the robot visits a cell with a coin, it always picks up that coin. Apply the dynamic programming algorithm in the book to find the total amount that the robot can pick up and the path that the robot should take. Show your steps to get the solution.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | 1 | 2 | 3 | 4 | 5 | 6 |
| 1 | 0 | 0 | **1**1 | **5**6 | 6 | 6 |
| 2 | 0 | **1**1 | 1 | **5**11 | **10**21 | 21 |
| 3 | **1**1 | 1 | **1**2 | 11 | 21 | **10**31 |
| 4 | 1 | 1 | **25**27 | 27 | **5**32 | 32 |
| 5 | 1 | **10**11 | 27 | **1**28 | 32 | 32 |

|  |  |  |
| --- | --- | --- |
| item | weight | value |
| 1 | 4 | $2 |
| 2 | 3 | $8 |
| 3 | 5 | $6 |
| 4 | 2 | $10 |

1. (**Bonus 50 points**) Apply the bottom-up dynamic programming algorithm to the following instance of the knapsack problem. Find the maximum value that can be put in the knapsack and which items should be selected. Show your steps to get the solution.

capacity

*Capacity j*

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | **0** | **1** | **2** | **3** | **4** | **5** | **6** |
| **0** | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| **1** | 0 | 0 | 0 | 0 | 2 | 2 | 2 |
| **2** | 0 | 0 | 0 | 8 | 8 | 8 | 8 |
| **3** | 0 | 0 | 0 | 8 | 8 | 8 | 8 |
| **4** | 0 | 0 | 10 | 10 | 10 | 18 | 18 |

{ max{F[i-1, j], vi + F[i-1, j- wi]} if j- wi >= 0

F[i, j] =

{ F[i – 1, j] if j- wi < 0

Initial conditions: F[0, j] = 0 and F[i, 0] = 0