# Quiz #2 CS361 Take home Spring 2017 Name: \_\_\_\_\_\_\_\_\_\_\_\_\_\_

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 0 | 64 | 184 | 1084 | 1384 | 1484 |
|  | 0 | 480 | 2760 | 2560 | 2920 |
|  |  | 0 | 1800 | 2400 | 2600 |
|  |  |  | 0 | 2250 | 3000 |
|  |  |  |  | 0 | 1500 |
|  |  |  |  |  | 0 |

**The quiz is open books, notes, and Internet. The quiz is NOT open to peers (currently in or not in our class)**

1. You are given the input p = <2, 8, 4, 15, 30, 5, 10>, populate the m,1 based and on the right, according to the MATRIC-CHAIN-ORDER(p) algorithm. Show your calculation for m[1,6], which should be 1484 (4 points).

1. Follow the 0/1 knapsack problem solution provided in class to solve the following problem: (6 points). The sack’s weight limit is 11. That is w = 18 (5 points).

|  |  |  |
| --- | --- | --- |
| Item | wi | vi |
| I1 | 4 | 7 |
| I2 | 3 | 6 |
| I3 | 5 | 9 |
| I4 | 2 | 4 |
| I5 | 1 | 5 |

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Item** | **0** | **1** | **2** | **3** | **4** | **5** | **6** | **7** | **8** | **9** | 10 | 11 |
| **1** | 0 | 0 | 0 | 0 | 7 | 7 | 7 | 7 | 7 | 7 | 7 | 7 |
| **2** | 0 | 0 | 0 | 6 | 7 | 7 | 7 | 13 | 13 | 13 | 13 | 13 |
| **3** | 0 | 0 | 0 | 6 | 7 | 9 | 9 | 13 | 15 | 16 | 16 | 16 |
| **4** | 0 | 0 | 4 | 6 | 7 | 10 | 11 | 13 | 15 | 17 | 19 | 20 |
| **5** | 0 | 5 | 5 | 9 | 11 | 12 | 15 | 16 | 18 | 20 | 22 | 24 |

1. If the problem described above is reclassified as a fractional knapsack one, the total value carried out by a sack of   
   capacity 20 should be \_\_\_\_\_\_\_ (3 points).
2. Show your Java/C++ code that can calculate Fibonacci number of 75, which is 2,111,485,077,978,050. We know fib(0) = 0 and fib(1) = 1. You cannot use recursive approach for this (5 points).
3. For the graph given, to run DFS starting from node 1, using the discovery time and finish time for each node (3 points).

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | 1 | 2 | 3 | 4 | 5 |
| d | 1 | 3 | 5 | 2 | 4 |
| f | 10 | 8 | 6 | 9 | 7 |

