Amanda Shohdy, Jasmine Rodgriguez, Edwin Rodriguez

Problem 1 Pseudocode and Efficiency Analysis

(10)Condense the daily act for both people down to one list called daily\_act

(1)Initialize a list and call it my\_list

(n)for i in range(len(person\_1\_schedule)):

(1)if person\_1\_schedule[i][0] < person\_2\_schedule[i][0]:

(1)Initialize a j to act as an index and set it equal to i

(1)Initialize a best\_time as a double and set it equal to 0.0

(n)while(person\_1\_schedule[i][1] < person\_2\_schedule[j][0]):

(1)if(person\_1\_schedule[i][1] < person\_2\_schedule[j][0]):

(1)best\_time = person\_2\_schedule[j][0]

(1)++j;

(1)my\_list.append({person\_1\_schedule[i][1], best\_time})

(1)else if person\_1\_schedule[i][0] > person\_2\_schedule[i][0]:

(1)Initialize a j to act as an index and set it equal to i

(1)Initialize a best\_time as a double and set it equal to 0.0

(n)while(person\_1\_schedule[i][0] > person\_2\_schedule[j][1]):

(1)if(person\_1\_schedule[i][0] > person\_2\_schedule[j][1]):

(1)best\_time = person\_2\_schedule[j][1];

(1)++j;

(1)my\_list.append({best\_time, person\_1\_schedule[i][0]})

(1)if person\_1\_schedule[i][1] > person\_2\_schedule[i][1]:

(1)Initialize a j to act as an index and set it equal to i

(1)Initialize a best\_time as a int and set it equal to 0

(n)while(j < len(person\_2\_schedule)):

(1)if person\_1\_schedule[i][1] < person\_2\_schedule[j][1]:

(1)best\_time = j

(1)++j

(1)if person\_1\_schedule[i][1] < person\_2\_schedule[best\_time][1]:

(1)my\_list.append(person\_1\_schedule[i][1], (1)person\_2\_schedule[best\_time][0])

(1)if my\_list[-1][0] > my\_list[-1][1];

(1)my\_list[-1][1] = daily\_act[1]

Step Count Analysis of Efficiency Class: 14n + 9n2 + 10 => O(n2)

Problem 2 Pseudocode and Efficiency Analysis

largestSum(V):

(1) b = 0, e = 1

(n) for i from 0 to n-1:

(n) for j from i+1 to n:

(1) if (sum(V[i:j] > sum(V[b:e]):

(1) b = i, e = j

(1) return (b, e)

Step Count Analysis of Efficiency Class: O(n2 + 4) => O(n2)