

**Instructions**

1. You are encouraged to work in groups of two. If you cannot find a partner, you can work alone.
2. Please submit **one** copy of the assignment; if you are working with a partner, both names should appear on the assignment.
3. **Part A** (your Java application) must be submitted by email to the TA (by the time of the lecture on the due date of the assignment). Note that you will get a zero if your program does not compile. **Part B** should be handed in at the beginning of the lecture on the due date.

**Part A**

Write a Java application which implements the dynamic programming solution to the activity selection problem. Let the program work as follows:

activityselection file.txt

where the format of file.txt is:

$$(s_1, f_1, p_1), (s_2, f_2, p_2), \dots, (s_n, f_n, p_n)$$

and the output is a list  $i_1, i_2, \dots, i_k$  of the subset of the  $n$  activities that optimizes the profit (where, of course,  $k \leq n$ ).

Before you do part A, do question 3 in part B.

**Part B**

1. Do problem 5.8 in the notes.

**Solution:** Consider  $w_1 = w_2 = 1$ ,  $w_3 = 2$  and  $C = 3$ , so the table on this input would look as follows:

	0	1	2	3
	T	F	F	F
$w_1 = 1$	T	T	F	F
$w_2 = 1$	T	T	T	F
$w_3 = 2$	T	T	T	T

Now consider the row for  $w_1 = 1$ , and the entry for the column labeled with 2. That entry is an F, as it should be, but if the for-loop in the algorithm were not a decreasing loop, then we would update that entry to a T since for  $j = 2$ , we have that  $2 \geq w_1$  and  $S(2 - w_1) = T$ .

2. Do problem 5.14 in the notes.

**Solution:**

$$\begin{cases} V(i-1, j) & \text{if } j < w_i \\ \max\{v_i + V(i-1, j-w_i), V(i-1, j)\} & \text{otherwise} \end{cases}$$

3. Do problem 5.15 in the notes.

**Solution:** The algorithm must include a computation of the distinct finish times, i.e., the  $u_i$ 's, as well as a computation of the array  $H$ . But here we just give the algorithm for computing  $A$  based on the recurrence—**this is the only part that you are responsible for**. The assumption is that there are  $n$  activities and  $k$  distinct finish times.

---

**Algorithm 1** Activity Selection

---

```

A(0) ← 0
for j : 1..k do
  max ← 0
  for i = 1..n do
    if  $f_i = u_j$  then
      if  $p_i + A(H(i)) > \text{max}$  then
        max ←  $p_i + A(H(i))$ 
      end if
    end if
  end for
  if  $A(j-1) > \text{max}$  then
    max ←  $A(j-1)$ 
  end if
  A(j) ← max
end for

```

---

4. Do problem 6.4 in the notes.

**Solution:** Pick  $1 \in V$ , and consider each  $(1, i') \in E$  in turn, remove it from  $G$  to obtain  $G_{1,i'} = ((V - \{1\}) \cup (V' - \{i'\}), E_{1,i'})$ , where  $E_{1,i'}$  consists of all the edges of  $E$  except those adjacent on 1 or  $i'$ , until for some  $i' \in V'$  we obtain a  $G_{1,i'}$  for which the algorithm answers “yes.” Then we know that there is a perfect matching that matches 1 and  $i'$ . Continue now with  $G_{1,i'}$ .