Duration: **75 minutes**
Aids Allowed: **NONE** (in particular, no calculator)

Student Number: |__|__|__|__|__|__|__|__|__|__|__|

Last (Family) Name(s): _____

First (Given) Name(s): _____

---

*Do **not** turn this page until you have received the signal to start.*
In the meantime, please read the instructions below *carefully*.

This term test consists of 2 questions on 8 pages (including this one), printed on both sides of the paper. *When you receive the signal to start, please make sure that your copy of the test is complete, fill in the identification section above, write your student number where indicated at the bottom of every odd-numbered page (except page 1), and write your name on the back of the last page.* Answer each question directly on the test paper, in the space provided.

The topic of this test is dynamic programming. All the questions have dynamic programming solutions and memorization solutions. When solving a question using dynamic programming *make sure you give a very clear definition of your array in english.* After that you should give the recurrence relation used to fill the array. These two together will account for most of the points of the question so make sure you get them right. Similarly, if you give a memorization solution make sure you clearly state (in english) what is the value your function is supposed to return. For example, "D[i] is the maximum profit we can make from cutting a rod of length i." or "optimalCut(i) will return the maximum profit we can make from cutting a rod of length i." Do not just jump to the (pseudo)code.

In your answers, you may use without proof any result or theorem covered in lectures, tutorials, homework, tests, or the textbook, as long as you give a clear statement of the result(s)/theorem(s) you are using. You must justify all other facts required for your solutions.

Write up your solutions carefully! In particular, use notation and terminology correctly and explain what you are trying to do.

If you are unable to answer a question, you will get 20% of the marks for that question if you write "I don't know" and nothing else — you will get 10% of the mark if your answer is completely blank. You will *not* get these marks if your answer contains contradictory statements (such as "I don't know" followed or preceded by parts of a solution that have not been crossed off).

MARKING GUIDE

1: _____/20

2: _____/10

TOTAL: _____/30

Good Luck!

OVER...

CONT'D...

Question 1. RLISP [20 MARKS]

RLISP (or Robot LISP) is a programming language designed to program robots. A RLISP program is just a string of 'R' and 'L's. The program is run by a robot who starts in the middle point (point 0) of the line shown in Figure 1 and reads the program taking one step left for every 'L' in the program and one step right for every 'R' in the program. For example if the program is "RLRLRRLLLR" then the robot will start at point 0 and go to points 1, 0, 1, 0, 1, 2, 1, 0, -1, 0 in that order finally stoping at point 0. The line the robot is walking on has length $2l$ so in a valid program the robot should never go beyond points $-l$ and l. Furthermore, at the end of any valid program the robot should end up at point 0 so that it is ready for the next person who wants to run a program on it. In other words a string $x[1 \dots n]$ is a valid program if and only if it has all the following properties:

1. It is made of only the characters 'R' and 'L',
2. The number of 'R' characters in x should be the same as the number of 'L' characters,
3. For any $1 \leq i \leq n$, the number of 'L' characters and 'R' characters in $x[1 \dots i]$ should be within l of each other.

Your task is to devise an algorithm that takes n, l as input and outputs the number of valid programs of length n. You can assume that $l \leq n$. To get the full marks your algorithm should run in time $O(n^2)$. Briefly justify the correctness and runtime of your algorithm.

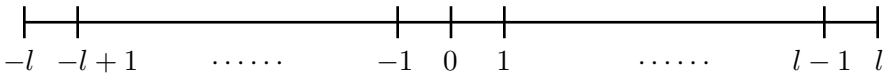

Figure 1: The line from Question 1.

Examples:

- For $n = 4$, $l = 2$ the answer is 6. The valid programs are "RRLL", "RLRL", "RLLR", "LRRL", "LRLR", "LLRR".
- For $n = 4$, $l = 1$ the answer is 4. The valid programs are "RLRL", "RLLR", "LRRL", "LRLR".
- For $n = 6$, $l = 2$ the answer is 18. The valid programs are "RRLRLL", "RRLLRL", "RRLLLR", "RLRRLL", "RLRLRL", "RLRLLR", "RLLRRL", "RLLRLR", "RLLLRR", "LRRRLL", "LRRLRL", "LRRLLR", "LRLRRL", "LRLRLR", "LRLLRR", "LLRRRL", "LLRRLR", "LLRLRR".
- For $n = 6$, $l = 1$ the answer is 8. The valid programs are "RLRLRL", "RLRLLR", "RLLRRL", "RLLRLR", "LRRLRL", "LRRLLR", "LRLRRL", "LRLRLR".

Hint: Don't try to give a closed form for the answer, use dynamic programming.

[There's more space on the next page...]

Question 1. (CONTINUED)

[Use this page for the rest of your solution.]

Question 2. The amusement park [10 MARKS]

It is 8 in the morning and you have just arrived at the amusement park. The park is a square and you are at the entrance at position $(0, 0)$ which is the lower left point of the park. There are n games in the park and you would like to play all of them, but unfortunately they are spread through out the park and each game starts at a particular time and ends at a later time. Specifically, the ith game is at position (x_i, y_i), starts at time t_i and finishes at time f_i. To play the ith game you have to be at point (x_i, y_i) at time t_i and stay there until time f_i when the game finishes. You can move around the park on foot and your walking speed is s, for example to get from point (x, y) to point (x', y') it will take you $\sqrt{(x - x')^2 + (y - y')^2}/s$ seconds.

Design an algorithm that takes the n, $(x_1, y_1), \ldots, (x_n, y_n)$, $t_1, \ldots, t_n$ and $f_1, \ldots, f_n$ and outputs the maximum number of games you can play. Briefly justify the correctness and runtime of your algorithm. To get the full marks your algorithm should run in time $O(n^2)$.

All times are in second since 8 in the morning, all distances are in meters and all speeds are in meters per second.

Examples:

- $n = 3$, $s = 2$, $(x_1, y_1) = (2, 4)$, $t_1 = 7$, $f_1 = 8$, $(x_2, y_2) = (4, 4)$, $t_2 = 3$, $f_2 = 6$, $(x_3, y_3) = (2, 2)$, $t_3 = 1$, $f_3 = 2$. Then the answer is 2. You walk from $(0, 0)$ to $(4, 4)$ and get there at time $\sqrt{8} \simeq 2.83$ and wait until time 3. You will then play game 2 until time 6. Afterwards you walk from $(4, 4)$ to $(2, 4)$ and get there at time 7 and play game 1 until time 8. It is impossible to play all three games simply because it is impossible to get to point $(2, 2)$ in time to play game 3.

- $n = 3$, $s = 1$, $(x_1, y_1) = (3, 4)$, $t_1 = 5$, $f_1 = 6$, $(x_2, y_2) = (4, 3)$, $t_2 = 8$, $f_2 = 10$, $(x_3, y_3) = (0, 5)$, $t_3 = 5$, $f_3 = 6$. Then the answer is 2. You walk from $(0, 0)$ to $(3, 4)$ and get there at time 5 just in time to play game 1 until time 6. Afterwards you walk from $(3, 4)$ to $(4, 3)$ and get there at time $6 + \sqrt{2} \simeq 7.41$ and wait until time 8 and play game 2 until time 10. It is impossible to play all three games because you can't play both games 1 and 3.

- $n = 3$, $s = 1$, $(x_1, y_1) = (3, 4)$, $t_1 = 5$, $f_1 = 6$, $(x_2, y_2) = (4, 3)$, $t_2 = 8$, $f_2 = 10$, $(x_3, y_3) = (0, 5)$, $t_3 = 16$, $f_3 = 17$. Then the answer is 3. You walk from $(0, 0)$ to $(3, 4)$ and get there at time 5 just in time to play game 1 until time 6. Afterwards you walk from $(3, 4)$ to $(4, 3)$ and get there at time $6 + \sqrt{2} \simeq 7.41$ and wait until time 8 and play game 2 until time 10. You will then walk to position $(0, 5)$ and get there at time $10 + \sqrt{20} \simeq 14.47$. You will then wait until time 16 and play game 3 until time 17.

[There's more space on the next page...]

Question 2. (CONTINUED)

[Use this page for the rest of your solution.]

On this page, please write nothing except your name.

Last (Family) Name(s): _____

First (Given) Name(s): _____