British Informatics Olympiad Final

31 March – 2nd April, 2006

Sponsored by Lionhead Studios

Robotics

This question is quite long — do not worry if you do not get to the end

In this question we will consider the motion of a robot arm in two dimensions, whose *base* is secured to a fixed position. Attached to the base is a chain of *links*, which are connected by *hinges*. These hinges can rotate through 360°. At the far end of the chain is the *hand*, and we are interested in positioning this hand by rotating the hinges. The following diagrams show how we can picture an arm:

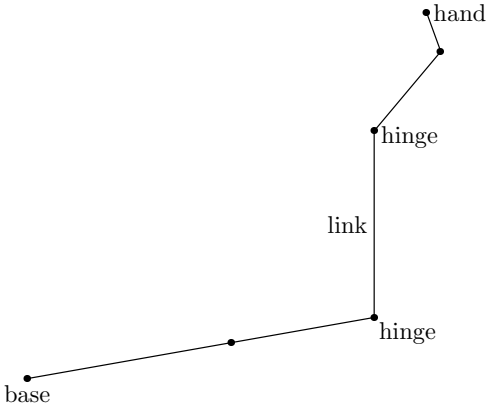


Figure I - an arm

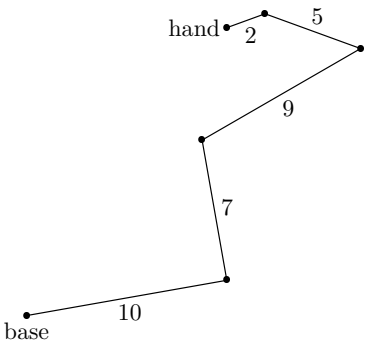


Figure II - the same arm, now moved

For the sake of this paper, you can consider that links can pass freely through each other. Thus the robot arm position depicted in figure III below is perfectly acceptable.

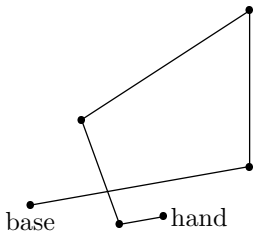


Figure III

When we give the lengths of the links of a robot arm, we always start from the base; so (for instance) we would say that the arm in figures I and II has lengths $10, 7, 9, 5$ and 2. The hinges, hand and base have negligible size and can be ignored.

The positions the hand can reach are limited by the lengths of the links. We do not only have problems reaching positions that are too far away. There may also be positions that cannot be reached because they are too close to the base.

Question 1

Consider an arm with two links of lengths 10 and 5. What is the farthest distance away from the base that this arm can reach? Some positions are too close to the base; what is the closest distance that it can still reach?

Question 2

Consider an arm with links of lengths 5 and 10. What is the farthest distance away it can reach? What is the closest distance that it can still reach?

Question 3

Now consider a robot arm with lengths l_1 and l_2. What is the farthest distance away it can reach? What is the closest distance that it can still reach?

Question 4

In fact, for any robot arm with two links, we can reach *any* distance that lies between the farthest distance away it can reach and the closest distance it can still reach. Why? (Only a brief explanation is necessary.)

Robot arms with only two links are quite boring. Let's now consider a robot arm with three links.

Question 5

What is the farthest distance an arm with links of length $100, 10$ and 8 can reach? What is the closest?

Question 6

What if the distances were $1, 10$ and 8? How about $3, 10$ and 8?

Question 7

Write pseudocode for a program which inputs three lengths l_1, l_2, l_3 and prints out the farthest and closest that a robot arm with links of those lengths can reach.

In fact, even with three or more links, we can reach any distance that is between the furthest distance away that it can reach and the closest distance it can still reach. You may use this fact freely from now on.

Question 8

Write pseudocode for a program which inputs a number n and n link lengths, and outputs the closest and furthest that a robot arm with those link lengths can reach. Your program should be recursive; that is, the version with n lengths should call the version with $n-1$ lengths.

In fact, there is a simpler program which computes the closest and furthest the arm can go. It looks like:

```
if A>0
    closest = 0
    farthest = B
else
    closest = C
    farthest = D
endif
```

where A, B, C and D are simple mathematical formulae involving only L (the length of the longest link) and T (the total length of the links).

Question 9
What should A, B, C and D be? Explain your answer.

Question 10
Suppose that you have a robot arm with link lengths $20, 3, 2$ and 1 and that we decide to only ever align links due north-south or east-west. How many different positions could the hand be in?

Question 11
If we changed the link lengths (keeping the 4 links), what is the largest number of different positions we could possibly be able to reach with the hand? What is the smallest?

How do we reach a position?

It is all well and good to know what we can reach, but sometimes it is useful also to know *how* we can reach it. Suppose that we have a robot arm with n links of length $l_1, l_2, \ldots, l_{n-1}, l_n$ and with base B, and we are trying to move the hand to a point P. It will turn out to be useful to consider our robot arm as consisting of one arm with $n-1$ links of lengths $l_1, l_2, \ldots, l_{n-1}$, with an extra single-link arm (of link length l_n) attached at the far end. We can draw the following diagram:

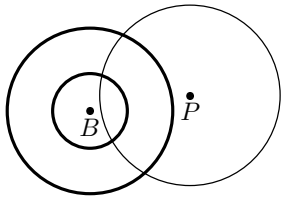


Figure IV

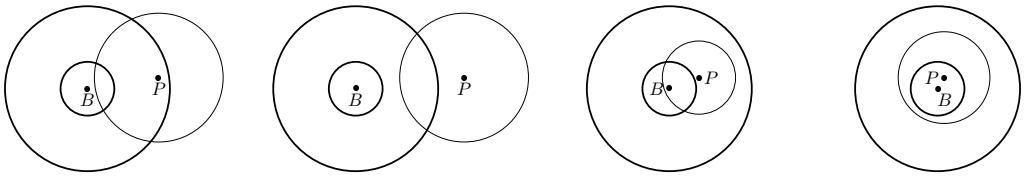
Here, we know that the $n-1$ link arm can reach all the points that are neither too close nor too far away, so we can picture the reachable area as the space between two circles (one of radius $r_{\min}$, one $r_{\max}$; these are the two bold circles on the diagram). We have also marked the point P, and a circle of radius l_n with P as its center. (This is the non-bold circle on the diagram.)

Question 12
Copy figure IV, and mark on it all the possible positions for the hinge between link $n-1$ and link n.

Question 13
Suppose that you knew the coordinates (x_B, y_B) of the base B, and the coordinates (x_P, y_P) of the point P in the above diagram, as well as l_n, $r_{\min}$ and $r_{\max}$. Suppose that the function `circle-intersect(`x_1`, `y_1`, `r_1`, `x_2`, `y_2`, `r_2`)` returns the coordinates of one of the points where the circle of radius r_1 center (x_1, y_1) meets the circle of radius r_2 center (x_2, y_2), or NONE if the circles do not intersect. Give example values which, when passed to this function, will return a possible pair of coordinates for the hinge between link $n-1$ and link n.

Question 14
Now write pseudocode for a general program which given any $n, l_1, \ldots, l_n$, (x_B, y_B) and (x_P, y_P), works out a possible position for the hinge before the nth link, if we are trying to reach point

P. You may use `circle-intersect`, any functions you have already written, and also a function `any-point(`x`, `y`, `r`)` which just returns *some* point on the circle center (x, y) radius r. You should carefully consider each of the following cases (but you may ignore the case when the hand cannot reach P):



Case 1 **Case 2** **Case 3** **Case 4**

Question 15

How could you use this to give a recursive algorithm that could determine a possible position for every hinge?

The two kinks theorem

Suppose we have an n link robot arm with more than 3 links, whose link lengths are $l_1, l_2, \ldots, l_n$. Now imagine marking a point on the arm that is exactly half way along it, when the arm is fully outstretched. The link which this point lies on is called the *median link*. (If the point we mark is at a hinge, then median link is the link connected to that hinge which is closest to the base.) Let's say that this is the m^{th} link.

Question 16

Write pseudocode for finding the median link.

The *two kinks theorem* states that if the original robot arm could reach a point, then so can the three-link robot arm with links of length $(l_1 + \cdots + l_{m-1}, l_m, l_{m+1} + \cdots + l_n)$. Thus it means that if a robot arm can reach a point, then it can do so with at most two *kinks*.

Question 17

Explain how the two-kink theorem can be used to create a non-recursive algorithm for placing the arm.

Question 18

Which of the two methods for positioning the arm do you think is better? Justify your answer.

Question 19

Give an explanation of why the two kinks theorem is true.