Hack the Technical Interview: Algorithms Practice

Presented By

Problem Three Interview Script: Pythagorean Triplets

Instructions

1. While the candidate reads the problem and prepares their answer, review this sheet in its entirety.
2. Mark the start and stop time so that you keep track.

Time Allowed: _____
Start Time: _____
Stop Time: _____

Problem Statement

Write a function that takes an array of elements and returns “Yes” if there is a pythagorean triplet ((a, b, c) that satisfies $a^2 + b^2 = c^2$), “No” otherwise.

```
01  ## input
02  triplet([3, 1, 4, 6, 5])
03
04  ## output
05  Yes
06
07  ## input
08  triplet([10, 4, 6, 12, 5])
09
10  ## output
11  No
12
```

Parameters

- The solution only needs to take into account unique integer arrays of length longer than 3.
- [Optional] You can choose to eliminate any built-in iterator methods in the language your candidate is using. This would include, for example, `itertool` library in Python. It's okay if you don't know enough about the language this candidate is using; you can skip this optional parameter.

Questions the Candidate Might Ask

Note: if the candidate asks a question that isn't answered here, feel free to make a decision and stick to that.

- Do the arrays contain repeated numbers?
  Answer: No. Each integer contained in the array is unique.

Questions the Interviewer Might Ask

Generic Questions

These are questions you could ask for any code challenge.

- What does this function need to do?
- How would you get started?
- What information do you need to store?

Guiding Questions

These are questions you can use to help the candidate if they get stuck.

- What computations will be repeated that you can cache?
  Answer: The square of the integers will be calculated for each comparison. Thus, calculating the square of each integer ahead of time will save computation time.
- How can you compare all triplet combinations of an array?
  Answer: Comparing the result of three for-loops is the easiest way to do this. Further efficiency improvements are covered in the hints below.

Hints

- Try looping over all possible combinations of numbers.
- Avoid repeated computations by saving/caching a frequently computed result.
- Try reducing the number of iterations using something similar to binary search on a sorted array.

If you give them all these hints and they still don't have a solution, you can start pointing them in the right direction with a line or two of code from one of the solutions below.

Solutions

Solution #1

```
def isTriplet(ar, n):
	j=0

	for i in range(n - 2):
		for k in range(j + 1, n):
			for j in range(i + 1, n - 1):
				x = ar[i]*ar[i]
				y = ar[j]*ar[j]
				z = ar[k]*ar[k]
				if (x == y + z or y == x + z or z == x + y):
					return 1
	return 0

def triplet(ar):
	if(isTriplet(ar, len(ar))):
		print("Yes")
	else:
		print("No")
```

Solution #2 (better)

```
def isTriplet(ar, n):
      for i in range(n):
            ar[i] = ar[i] * ar[i]
      ar.sort()
for i in range(n-1,1,-1):
      j = 0
      k = i - 1
      while (j < k):
            if (ar[j] + ar[k] == ar[i]):
                  return true
            else:
                  if (ar[j] + ar[k] < ar[i]):
                        j = j + 1
                  else:
                        k = k - 1
                  return False

def triplet(ar):
      if (isTriplet(ar, len(ar))):
            print ("Yes")
      else:
            print ("No")
```