**Question #1:**

Given a sorted array, write an algorithm to find out if a given number is in that array. Assume that all numbers in the array are integers. Your solution should be O(log2N). There are two solutions with this time complexity.

Output should be Boolean

*# Recursive approach*

**def** **binarySearch**(sorted\_arr:*list*, l:*int*, r:*int*, x:*int*)->bool:

if r >= l:

*mid* **=** l **+** (r **-** l) **//** 2

# If element is present at the middle itself

if sorted\_arr[mid] == x:

return True

# If element is smaller than *mid*, then it

# can only be present in left subarray

elif sorted\_arr[mid] > x:

return binarySearch(*sorted\_arr*, *l*, mid-1, *x*)

# Else the element can only be present

# in right subarray

else:

return binarySearch(*sorted\_arr*, mid + 1, *r*, *x*)

else:

# Element is not present in the array

return False

*# Iterative approach*

**def** **binarySearch**(arr:*list*, l:*int*, r:*int*, x:*int*)->bool:

while l <= r:

*mid* **=** l **+** (r **-** l) **//** 2;

# Check if x is present at mid

if arr[mid] == x:

return True

# If x is *greater*, ignore left half

elif arr[mid] < x:

*l* **=** mid **+** 1

# If x is *smaller*, ignore right half

else:

*r* **=** mid **-** 1

# If we reach *here*, then the element

# was not present

return False

**Question #2:**

Given a DNA sequence (A string that only contains A, C, T and G), find the beginning of a gene and return the amino sequence of the gene.

* A codon is three nucleotides (letters) together. For example ‘AAA’ is a codon
* The beginning of a gene is is marked by the start codon: ‘ATG’
* Once you have found the start codon, the second codon is the nucleotides 4,5 and 6. For example in the DNA sequence: ‘GGGATGAAA’ the second codon is ‘AAA’
* You can calculate the amino acid sequence by matching each codon to an amino acid in a codon table
* Assume you have the codon table stored as a dictionary in memory

**def** **DNA2protein**(s:*str*)->str:

*codon\_table* **=** {

'ATA':'I', 'ATC':'I', 'ATT':'I', 'ATG':'M',

'ACA':'T', 'ACC':'T', 'ACG':'T', 'ACT':'T',

'AAC':'N', 'AAT':'N', 'AAA':'K', 'AAG':'K',

'AGC':'S', 'AGT':'S', 'AGA':'R', 'AGG':'R',

'CTA':'L', 'CTC':'L', 'CTG':'L', 'CTT':'L',

'CCA':'P', 'CCC':'P', 'CCG':'P', 'CCT':'P',

'CAC':'H', 'CAT':'H', 'CAA':'Q', 'CAG':'Q',

'CGA':'R', 'CGC':'R', 'CGG':'R', 'CGT':'R',

'GTA':'V', 'GTC':'V', 'GTG':'V', 'GTT':'V',

'GCA':'A', 'GCC':'A', 'GCG':'A', 'GCT':'A',

'GAC':'D', 'GAT':'D', 'GAA':'E', 'GAG':'E',

'GGA':'G', 'GGC':'G', 'GGG':'G', 'GGT':'G',

'TCA':'S', 'TCC':'S', 'TCG':'S', 'TCT':'S',

'TTC':'F', 'TTT':'F', 'TTA':'L', 'TTG':'L',

'TAC':'Y', 'TAT':'Y', 'TAA':'\_', 'TAG':'\_',

'TGC':'C', 'TGT':'C', 'TGA':'\_', 'TGG':'W',

}

# First find the start codon and extract the sequence of the gene

*current\_codon* **=** '...'

*end\_codon* **=** '...'

*coding\_seq* **=** ''

for *i*,j in enumerate(*s*):

current\_codon **+=** j

current\_codon **=** current\_codon[1:]

**if** current\_codon **==** 'ATG':

coding\_seq **=** 'ATG'

**for** nuc **in** s[i:]:

coding\_seq **+=** nuc

**if** **len**(coding\_seq) **%** 3 **==** 0:

end\_codon **=** coding\_seq[**-**3:]

**if** end\_codon **in** ['TGA','TAG','TAA']:

**break**

**break**

*# Now convert the coding sequence to protein*

protein\_seq **=** ''

**for** starting\_nuc **in** **range**(0, **len**(coding\_seq),3):

codon **=** coding\_seq[starting\_nuc:starting\_nuc**+**3]

protein\_seq **+=** codon\_table[codon]

**return** protein\_seq

**SQL**

**Question #1**

Select the *ProductName*, *Inventory* from the Products table and create a new column called *Reorder* which has a value ‘Yes’ if a product’s *Inventory* is below 50, and ‘No’ if it is equal to or above 50.

SELECT

ProductName, Inventory,

CASE WHEN Inventory >= 50 THEN ‘No’ ELSE ‘Yes’ END AS Reorder

FROM

Products;

**Question #2**

From **Employees**, return the *Department*, *EmployeeName* and *Salary* for the employee with the highest salary in each department.

SELECT

Department, EmployeeName, Salary

FROM

Employees

WHERE

(Department , Salary) IN

( SELECT

Department, MAX(Salary)

FROM

Employee

GROUP BY Department

);