ASSIGNMENT- 1

1. Use Mathematical Induction to prove :-
For any natural numbers, $n^3 - n$ is divisible by 3

→ Let $n = 1$; $n^3 - n = 1^3 - 1 = 0$ => divisible by 3 = 3(0)
$n = 2$; $n^3 - n = 2^3 - 2 = 8 - 2 = 6$ => divisible by 3 = 3(2)

According to mathematical induction,
assume the above will be true for $n = k$
Let $n = k$; $n^3 - n = k^3 - k = 3(m)$ .... which is divisible by 3
→ (1)

Let $n = k+1$, if $n = k$ is true then $n = k+1$ will be true
$(n^3 - n) = (k+1)^3 - (k+1) = k^3 + 3k^2 + 3k + 1 - k - 1$
$= k^3 + 3k^2 - 2k$
$= (k^3 - k) + 3(k^2 + k)$

∴ Substituting $k^3 - k = 3m$ from (1),
$= 3m + 3(k^2 + k)$
$= 3(m + k^2 + k)$
..... which is divisible by 3.

∴ According to the above proof of mathematical induction,
$n^3 - n$ is divisible by 3.

2. Write algorithm of calculating Factorial by following the requirements :-
a) Write the algorithm using loop (as oppose to recursion)
→ Algorithm for finding factorial :-
① Start
② Declare variable n, i, factorial
③ Read number n from user
④ Initialise variable factorial = 1 and i = 1
⑤ Repeat until i <= n
then factorial = factorial * i
i = i + 1

⑥ Print factorial
⑦ Stop

Pseudo code for calculating factorial:-

```
read n
factorial = 1
i = 1
WHILE i <= n
    factorial = factorial * i
    i = i + 1
END WHILE
WRITE factorial
```

b) Explain what "loop invariant" is and prove the algorithm is correct.

→ Loop invariant is a condition that is true each time the loop test is evaluated. It is used to check the correctness of an algorithm. A loop variant is a good one when it satisfies these three properties:-

① Initialization
② Maintenance
③ Termination.

Proving correction using mathematical induction:-

For any natural number n,

$n! = n * (n-1) * (n-2) * \ldots * 3 * 2 * 1 = n * (n-1)!$

Let n = 1, 2, 3

$n! = 1! = 1 * (1-1) = 1$

$n! = 2! = 2 * (2-1) = 2 * 1 = 2$

$n! = 3! = 3 * (3-1) = 3 * 2 = 6$

According to mathematical induction, from the previous steps, assume, this will be true for $n = k$.

Let $n = k$.

$n! = k! = k*(k-1)*\dots*2*1 = k*(k-1)!$ ⟶ (1)

Let $n = k+1$, if $n = k$ is true then $n = k+1$ will be true,

$n! = (k+1)! = (k+1)*k*(k-1)*\dots*2*1$

$= (k+1)*k!$ ..... Substituted $k! = k*(k-1)*\dots*2*1$ from eq. (1)

∴ According to above proof of mathematical induction, $n! = n*(n-1)!$ is true.

c) Calculate the running time of this algorithm (you can assume any constant)

→ $T(n) = \theta(1) + \theta(1) + \theta(1) + \theta(n) + \theta(n) + \theta(n)$

$= \theta(3n) = \theta(n)$

Explanation :-

read n	$\theta(1)$
factorial = 1	$\theta(1)$
i = 1	$\theta(1)$
WHILE i <= n	$\theta(n)$
factorial = factorial * i	$\theta(n)$
i = i + 1	$\theta(n)$

d) Write algorithm using recursion.

→
```
read n
if n = 1
    return 1
return n * fact (n-1)
```

3. Solve the "Maximum Ascending Subarray Sum" problem (no, 1800) on leetcode.com, then follow the requirements:

a. Make a successful submission. Attach the screenshot image of your submission
Note. your submission image should be something like this (that is, pass all test cases, instead of an accept page of a particular test case)

Success Details ›

Runtime: 0 ms, faster than 100.00% of Java online submissions for Maximum Ascending Subarray Sum.

Memory Usage: 36.6 MB, less than 63.64% of Java online submissions for Maximum Ascending Subarray Sum.

Next challenges:

Find Good Days to Rob the Bank

Show off your acceptance:

Time Submitted	Status	Runtime	Memory	Language
01/07/2022 16:23	Accepted	0 ms	36.6 MB	java