Assignment 1

Ch Pranay Prakash - EE18BTECH11009

Download all C codes from

```
https://github.com/pranayEE11009/
    C_and_DataStructures/tree/main/
    Assignment_1/codes
```

and latex-tikz codes from

```
https://github.com/pranayEE11009/
    C_and_DataStructures/tree/main/
    Assignment_1
```

1 Problem

Consider the following ANSI C function:

```
int SimpleFunction(int Y[], int n, int x)
{
int total = Y[0], loopIndex;
for (loopIndex=1; loopIndex<=n−1; loopIndex++)
    {
    total=x∗total +Y[loopIndex];
    }
return total;
}
```

Let Z be an array of 10 elements with Z[i]=1, for all i such that 0≤i≤9. The value returned by SimpleFunction(Z,10,2) is ?

2 Solution

Solution: 1023

Code to generate the solution:

```
#include <stdio.h>
int SimpleFunction(int Y[], int n, int x){
    int total = Y[0], loopIndex;
    for( loopIndex = 1; loopIndex<=n−1;
        loopIndex++){
        total = x∗total + Y[loopIndex];
    }return total;
}
int main()
{
    int Z[10] = {1,1,1,1,1,1,1,1,1,1};
    printf("%d", SimpleFunction(Z, 10, 2));
    return 0;
}
```

The function **S**impleFunction of the C code in the question takes an integer type array (Y[]), and two integer variables (n and x) as the inputs and returns an integer as the output.

The inputs of the SimpleFunction are:

1) integer type array, Z[i] = 1 for all 0≤i≤9 i,e., Z = [1,1,1,1,1,1,1,1,1,1]
2) integer n = 10
3) integer x = 2

In the function SimpleFunction(Z,n,x) a "for loop" is run for n-1 iterations.
In each iteration the integer variable "**total**", which is initiated with 1, is recursively multiplied with 2 with its previous value and added to 1.

```
    total=x∗total + Z[loopIndex];
```

Since, Z[i] is always 1 and $x = 2$.

```
    total=2∗total + 1;
```

The values of total for n-1 iterations are,
initially total = 1
for loopIndex = 1, total = 2*(1) + 1 = 3
for loopIndex = 2, total = 2*(3) + 1 = 7
for loopIndex = 3, total = 2*(7) + 1 = 15
for loopIndex = 4, total = 2*(15) + 1 = 31
for loopIndex = 5, total = 2*(31) + 1 = 63
for loopIndex = 6, total = 2*(63) + 1 = 127
for loopIndex = 7, total = 2*(127) + 1 = 255
for loopIndex = 8, total = 2*(255) + 1 = 511
for loopIndex = 9, total = 2*(511) + 1 = 1023

The for loop terminates at loopIndex = 9, and the SimpleFunction returns the final value of total, which is equal to **1023**.

Now, as we observe the values of ”total” (3,7,15...,1023), we can observe that each value of ”total” is one less than some integer exponential of 2. For example;

for loopIndex = 1, total = 3 = $2^2 - 1$
for loopIndex = 2, total = 7 = $2^3 - 1$ and so on.

So, lets take a general equation of total,

$$\boxed{T(m) = 2T(m-1) + 1}$$

where m is the iterative loopIndex of the for loop

Now,

$$\begin{aligned} T(m) &= 2T(m-1) + 1 \\ T(m-1) &= 2T(m-2) + 1 \\ T(m-2) &= 2T(m-3) + 1 \\ &\vdots \\ T(2) &= 2T(1) + 1 \end{aligned}$$

$\because loopIndex \geq 1, m \geq 1$
Now, to get a general solution to the above equations we multiply each equation with suitable coefficients and add them,

$$\begin{aligned} T(m) &= 2T(m-1) + 1 \\ 2T(m-1) &= 2 * (2T(m-2) + 1) \\ 2^2 * T(m-2) &= 2^2 * (2T(m-3) + 1) \\ &\vdots \\ 2^{m-2} * T(2) &= 2^{m-2} * (2T(1) + 1) \end{aligned}$$

Now, on adding all the above equations we get rid of all the T(m-i) form values except T(m), T(1) and left with all the 1’s from each equation,

$$\begin{gathered} T(m) = 1 + 2 + 2^2 + 2^3 + ... + 2^{m-2} + 2^{m-1}T(1) \\ T(m) = 2^{m-1} - 1 + 2^{m-1} * 3 \\ [\because T(1) = 2 * (1) + 1 = 3] \\ T(m) = 4 * 2^{m-1} - 1 \\ T(m) = 2^{m+1} - 1 \end{gathered}$$

We can find ”total” value using the equation below, where m is the iterative index ”loopIndex”.

$$T(m) = 2^{m+1} - 1 \tag{2.0.1}$$

Since the SimpleFunction returns the total value, which corresponds for loopIndex = n-1.
So, for m = n-1, we have

$$T(n-1) = 2^n - 1 \tag{2.0.2}$$

Finally, we can find the final output of the SimpleFunction using the above equation for different values of n.

In the question, the value of n=10,

$$\implies total = 2^{10} - 1 = \mathbf{1023}$$

Note: *The above equation is only for x = 2 and Z = [1,1,1,1,1,1,1,1,1,1]*