Assignment 1

Sudeep Veggalam - EE18BTECH11045

Download all latex-tikz codes from

https://github.com/sudeepv/EE4013/blob/main/Assignment1/assignment1.tex

1 Problem

Consider the following C program.

```
#include <stdio.h>
int main()
{
    int a[] = {2,4,6,8,10};
    int i,sum = 0, *b = a + 4;
    for (i = 0; i < 5; i++)
        sum = sum + (*b - i) - *(b - i);
    printf("%d\n", sum);
    return 0;
}
```

The output of the above C program is?

2 Solution

Output : 10

Explanation

This is a problem involving pointers. As in C, the array variable is considered to be a pointer pointing to the first array element. The following part of the code initializes the pointer b to a[4]. Since, int $*b = a+4 \implies$ implies b points to a[0+4] = a[4].

```
int *b = a + 4;
```

In every iteration of the loop, the value of $(*b - i) - *(b - i)$ is added to the variable sum. This is implemented in the line below:

```
sum = sum + (*b - i) - *(b - i);
```

Initially i = 0, count = 0. Since the terminating condition is $i < 5$, the loop runs from i = 1 to i = 4.The updating process in the loop is shown in the table below. The following line outputs the value of sum as the loop ends.

```
printf("%d\n", sum);
```

i	*b-i	*(b-i)	*b-i - *(b-i)	sum
1	9	8	1	1
2	8	6	2	3
3	7	4	3	6
4	6	2	4	10

The value of sum = 10. Hence, output of the program is 10.

Code without using pointers The main usage of pointers in the given code is by creating a pointer $*b = a + 4$ which stores the address of the array element a[4]. Instead, we can create another variable int b = 4 which stores the index of the array element instead of it's address. The modified code is given below:

```
#include <stdio.h>
int main()
{
    int a[] = {2,4,6,8,10};
    int i,sum = 0, b = 4;
    for (i = 0; i < 5; i++)
        sum = sum + (arr[b] - i) - arr[b-i];
    printf("%d\n", sum);
    return 0;
}
```

As we are storing the index of the element in b instead of the pointer, we use $arr[b-i]$ to access the element instead of using $*(b-i)$. In the code, $(*b-i)$ is modified as $arr[b]-i$ and $*(b-i)$ is modified as $arr[b-i]$. The iteration process is shown in the table below: From the table, the answer is 10.

i	arr[b]-i	arr[b-i]	arr[b]-i - arr[b-i]	sum
1	9	8	1	1
2	8	6	2	3
3	7	4	3	6
4	6	2	4	10