Assignment 1(unit1-Github and arrays):

Cpp code

Naive approach:

```cpp
#include <bits/stdc++.h>
using namespace std;

int main()
{
    int arr[] = {1, 5, 8, 9, 6, 7, 3, 4, 2, 0};
    int n = sizeof(arr) / sizeof(arr[0]), target = 6;
    for (int i=0; i < n; i++)
    {
        for(int j=i;j<n;j++){
            if (arr[i] + arr[j] == target and i!=j)
            {
                cout << i << "  " << j<<"\n";
                return 0;
            }
        }
    }
    return 0;
}
```

2 pointers:

```cpp
#include <bits/stdc++.h>
using namespace std;

int main()
{
    int arr[] = {1, 5, 8, 9, 6, 7, 3, 4, 2, 0};
    int n = sizeof(arr) / sizeof(arr[0]), target = 6;
    sort(arr, arr + n);
    int i = 0, j = n - 1;
    while (i < j)
    {
        if (arr[i] + arr[j] == target)
        {
            cout << i << "  " << j;
            break;
```

```
        }
        else if (arr[i] + arr[j] < target)
        {
            i++;
        }
        else
        {
            j--;
        }
    }
    return 0;
}
```

Hashmap:

```
#include <bits/stdc++.h>
using namespace std;

int main()
{
    int arr[] = {3, 2, 4};
    int n = sizeof(arr) / sizeof(arr[0]), target = 6;
    map<int, int> mp;
    int i = 0;
    for (int i = 0; i < n; i++)
    {
        if (mp[target - arr[i]] == 0)
        {
            mp[target - arr[i]] = i + 1;
        }
        if (mp[arr[i]] && i != 0)
        {
            cout << mp[arr[i]] - 1 << "," << i << endl;
            break;
        }
    }
    return 0;
}
```

- Test cases passed
- Completed on 11/3/23

Q/A:

1. How long did you spend on this assignment?
    a. 1day
2. Based on your effort, what letter grade would you say you earned?
    a. On a scale of 1 to 10. I would grade this as 10/10.
3. Based on your solution, what letter grade would you say you earned?
    a. On a scale of 1 to 10. I would grade this as 9/10.
4. Provide a summary of what doesn't work in your solution, along with an explanation of how you attempted to solve the problem and where you feel you struggled?
    a. Naive approach is to check all possible combinations which is O(n2).
    b. Optimising the time complexity using 2 pointer approach with initial sorting which takes O(n log n)
    c. Using hashmap by finding the remaining value which (target-array[i]) as a key and index as value which helps us to search the remaining value exists in the previous state. The complexity is O(n) and space O(n)