Week 6

Problem 1:

You are given a sequence of n-1 distinct positive integers, all of which are less than or equal to an integer 'n'. You have to find the integer that is missing from the range [1,2, ... n]. Solve the question without using arrays.

Answer: (penalty regime: 0 %)

```
#include<stdio.h>
int main()
{
    int n,num,sum=0,tot;
    scanf("%d",&n);
    tot=n*(n+1)/2;
    for(int i=0;i<n-1;i++)
    {
        scanf("%d",&num);
        sum+=num;
    }
    printf("%d\n",tot-sum);
    return 0;
}
```

	Input	Expected	Got	
✓	3 1 2	3	3	✓
✓	4 1 3 4	2	2	✓

Passed all tests! ✓

Result : The program successfully prints the missing number .

Problem 2:

A Teacher came to the class with a large box that has several coins. Each coin has a number Printed on it. Before Coming to the class, she ensured that all the coins occurs an even number of times. However, while coming to the class one coin fell down and got lost. She wants to find out the number of missing coin (Solve the question without using arrays).

```c
#include<stdio.h>
int main()
{
    int n,num,res=0;
    scanf("%d",&n);
    for(int i=0;i<n-1;i++)
    {
        scanf("%d",&num);
        res^=num;
    }
    printf("%d\n",res);
    return 0;
}
```

	Input	Expected	Got	
✓	8 5 7 2 7 5 2 5	5	5	✓
✓	6 5 5 6 6 6	6	6	✓

Passed all tests! ✓

Result : The programs successfully prints the missing coin.

Problem 3:

An abundant number is a number for which the sum of its proper divisors is greater than the number itself.

Proper divisors of the number are those that are strictly lesser than the number.

Input Format:

Take input an integer from stdin

Output Format:

Print Yes if given number is Abundant. Otherwise, print No

```c
#include<stdio.h>
int main()
{
    int n,sum=0;
    scanf("%d",&n);
    for(int i=1;i<n;i++)
    {
        if(n%i==0)
        sum+=i;
    }
    if(sum>n)
    printf("Yes");
    else
    printf("No");
    return 0;
}
```

	Input	Expected	Got	
✓	12	Yes	Yes	✓
✓	13	No	No	✓

Passed all tests! ✓

Result : The program successfully identifies the abundant number.

Problem 4:

Linear search -

```c
#include<stdio.h>
int main()
{
    int n,found=0;
    scanf("%d",&n);
    int i,a[n];
    for(i=0;i<n;i++)
    {
        scanf("%d",&a[i]);
    }
    int search_element;

    scanf("%d",&search_element);
    for(i=0;i<n;i++)
    {
        if(search_element==a[i])
        {
            printf("Element found at location : %d",i);
            found=1;
        }
    }
    if(found==0)
    printf("Element not found.");
    return 0;

}
```

	Input	Expected	Got	
✓	5 30 40 50 20 10 20	Element found at location : 3	Element found at location : 3	✓
✓	5 30 40 50 20 10 55	Element not found.	Element not found.	✓

Result : Hence the element was found using linear search.

Problem 5 : Binary search

```c
#include<stdio.h>
int main()
{
    int n;
    scanf("%d",&n);
    int ar[n];
    for(int i=0;i<n;i++)
    scanf("%d",&ar[i]);
    int a;
    scanf("%d",&a);
    int low=0;
    int high=n-1;
    int mid=(low+high)/2;
    while(low<=high)
    {
        if (ar[mid]<a)
        low=mid+1;
        else if(ar[mid]==a)
        {
            printf("Element found at location : %d",mid);
            break;
        }
        else
        high=mid-1;
        mid=(low+high/2);
    }
    if(low>high)
    printf("Element not found.");
    return 0;
}
```

	Input	Expected	Got	
✓	5 10 20 30 40 50 30	Element found at location : 2	Element found at location : 2	✓

Result : The element was search using binary search.

Problem 6: Insertion sort

```c
#include<stdio.h>
int main()
{
    int n,i,j,temp;
    scanf("%d",&n);

    int a[n];
    for(i=0;i<n;i++)
    scanf("%d",&a[i]);
    for(i=1;i<=n-1;i++)
    {
        for(j=i;j>0&&a[j-1]>a[j];j--)
        {
            temp=a[j];
            a[j]=a[j-1];
            a[j-1]=temp;
        }
    }
    for(i=0;i<n;i++)
    printf("%d ",a[i]);
    return 0;
}
```

	Input	Expected	Got	
✓	5 30 40 50 20 10	10 20 30 40 50	10 20 30 40 50	✓

Passed all tests! ✓

Result : The array was then sorted using insertion sort.

Problem 7:

Selection sort

```c
#include<stdio.h>
int main()
{
    int n,i,j,temp;
    scanf("%d",&n);
    int a[n];
    for(i=0;i<n;i++)
    scanf("%d",&a[i]);
    for(i=0;i<n;i++)
    {
        for(j=i+1;j<n;j++)
        {
            if(a[i]>a[j])
            {
                temp=a[i];
                a[i]=a[j];
                a[j]=temp;
            }
        }
    }
    for(i=0;i<n;i++)
    printf("%d ",a[i]);
    return 0;
}
```

	Input	Expected	Got	
✓	5 30 40 50 20 10	10 20 30 40 50	10 20 30 40 50	✓

Passed all tests! ✓

Result : The array was then sorted using selection sort.

Problem 8: Bubble sort

```c
#include<stdio.h>
int main()
{
    int n,i,j,flag=0;
    scanf("%d",&n);
    int a[n];
    for(i=0;i<n;i++)
    scanf("%d",&a[i]);
    for(i=0;i<n-1;i++)
    {
        flag=0;
        for(j=0;j<n-1-i;j++)
        {
            if(a[j]>a[j+1])
            {
                int temp=a[j];
                a[j]=a[j+1];
                a[j+1]=temp;
                flag=1;
            }
        }
        if(flag==0)
        break;

    }
    for(i=0;i<n;i++)
    printf("%d ",a[i]);
    return 0;
}
```

	Input	Expected	Got	
✓	5 30 40 50 20 10	10 20 30 40 50	10 20 30 40 50	✓

Result : The array was then sorted using bubble sort.

Problem 9:

Quick sort

Answer: (penalty regime: 0 %)

```
    *a = *b;
    *b=temp;
}

// partition function
int partition(int a[], int low, int high){
    int pivot = a[high];
    int i=low-1;
    for(int j=low;j<high;j++){
        if(a[j]<=pivot){
            i++;
            swap(&a[i],&a[j]);
        }
    }
    swap(&a[i+1],&a[high]);
    return i+1;
}

// quick sort
void quickSort(int a[],int low,int high){
    if(low<high){
        int pi = partition(a,low,high);
        quickSort(a,low,pi-1);
        quickSort(a,pi+1,high);
    }
}

int main(){
    int size;
    scanf("%d",&size);
    int a[size];
    for(int i=0;i<size;i++){
        scanf("%d",&a[i]);
    }
    quickSort(a,0,size-1);
    for(int i=0;i<size;i++){
        printf("%d ",a[i]);
    }
    return 0;
}
```

	Input	Expected	Got	
✓	5 30 40 50 20 10	10 20 30 40 50	10 20 30 40 50	✓

Passed all tests! ✓

Result : The array was then sorted using quick sort.

Problem 10 :

Merge sort

```c
#include<stdio.h>
void merge(int a[],int l,int m,int r)
{
    int n1=m-l+1,n2=r-m,L[n1],R[n2];
    for(int i=0;i<n1;i++)
    L[i]=a[l+i];
    for(int i=0;i<n2;i++)
    R[i]=a[m+1+i];
    int i=0,j=0,k=l;
    while(i<n1&&j<n2)
    a[k++]=(L[i]<=R[j])?L[i++]:R[j++];
    while(i<n1)
    a[k++]=L[i++];
    while(j<n2)
    a[k++]=R[j++];
}
void mergesort(int a[],int l, int r)
{
    if(l<r)
    {
        int m=l+(r-l)/2;
        mergesort(a,l,m);
        mergesort(a,m+1,r);
        merge(a,l,m,r);
    }
}
int main()
{
    int a[100],n;
    scanf("%d",&n);
    for(int i=0;i<n;i++)
    scanf("%d",&a[i]);
    mergesort(a,0,n-1);
    for(int i=0;i<n;i++)
    printf("%d ",a[i]);
    return 0;
}
```

	Input	Expected	Got	
✓	5 30 40 50 20 10	10 20 30 40 50	10 20 30 40 50	✓

Passed all tests! ✓

Result : The array was then sorted using quick sort.

-------X-------X-------X-------X-------X-------