Data Structures and algo in java - Day 16

Learned about Kadane’s Algorithm to find maximum Sub Array sum in an array

Did a problem on Arranging the value in Positive and negative order.

Learned about Permutations , the concept is a bit confusing but got a hang of it

Here are the Sums I have solved today

import java.util.\*;

public class day16

{

public static void main(String[] args)

{

//int arr [] = {9,9,4,4,3,3,7,0,0};

// int arr [] = {-2,-3,4,-1,-2,1,5,-3};

// int arr [] = {-1,-2,2,3,-1,2,-3,-9,-20,2,1,3,7,2};

int arr [] = {1,2,3};

permutation(arr);

}

public static int kadane(int arr[])

{

int n = arr.length;

int sum = 0;

int max = Integer.MIN\_VALUE;

for(int i=0;i<n;i++)

{

sum = sum + arr[i];

if(sum>max)

{

max = sum;

}

if(sum<0)

{

sum=0;// no use of carrying negative numbers in the sum

}

}

return max;

}

public static void kadaneArray(int arr[])

{

int n=arr.length;

int sum = 0;

int ansStart = -1;

int ansEnd =-1;

int start = 0;

int max=Integer.MIN\_VALUE;

for(int i=0;i<n;i++)

{

if(sum==0)

{

start = i;

}

sum = sum+arr[i];

if(sum>max)

{

max=sum;

ansStart= start;

ansEnd =i;

}

if(sum<0)

{

sum=0;

}

}

for(int i=ansStart;i<=ansEnd;i++)

{

System.out.print(arr[i]+" ");

}

System.out.println("The Maximum Sum is: "+max);

}

public static void reArrange(int arr[])

{

int n = arr.length;

int positive =0;

int negative =1;

int result [] = new int[n];

for(int i=0;i<n;i++)

{

if(arr[i]>0)

{

result[positive]=arr[i];

positive+=2;

}

else if(arr[i]<0)

{

result[negative]=arr[i];

negative+=2;

}

}

System.out.println(Arrays.toString(result));

}

public static void reArrangeUnEven(int arr [])

{

int n = arr.length;

ArrayList<Integer> positive = new ArrayList<>();

ArrayList<Integer> negative = new ArrayList<>();

for(int i=0;i<n;i++)

{

if(arr[i]>0)

{

positive.add(arr[i]);

}

else

{

negative.add(arr[i]);

}

}

if(positive.size()>negative.size())

{

for(int i=0;i<negative.size();i++)

{

arr[i\*2] = positive.get(i);

arr[i\*2+1] = negative.get(i);

}

int index = negative.size()\*2;

for(int i=negative.size();i<positive.size();i++)

{

arr[index]=positive.get(i);

index++;

}

}

if(negative.size()>positive.size())

{

for(int i=0;i<positive.size();i++)

{

arr[i\*2] = positive.get(i);

arr[i\*2+1] = negative.get(i);

}

int index = positive.size()\*2;

for(int i=positive.size();i<negative.size();i++)

{

arr[index]=negative.get(i);

index++;

}

}

System.out.println(Arrays.toString(arr));

}

public static void permutation(int arr[])//recieves the array to permutate

{

List<List<Integer>> ans = new ArrayList<>();//creates the answer list of list

recursionPermutation(0,arr,ans);//sends the index value,array and the list of list of ans to be stored

System.out.println(ans);//prints the ans

}

public static void recursionPermutation(int index,int arr [], List<List<Integer>> ans) //recieves the things

{

if(index==arr.length) // if the index is equal to the length then get the data from array and add to the answer

{

List<Integer> data = new ArrayList<>();

for(int i=0;i<arr.length;i++)

{

data.add(arr[i]);

}

ans.add(data);

return;

}

for(int i=index;i<arr.length;i++)

{

swap(i,index,arr);

recursionPermutation(index+1, arr, ans);

swap(i,index,arr);

}

}

public static void swap(int i, int j, int arr[])

{

int temp = arr[i];

arr[i]= arr[j];

arr[j] = temp;

}

}