Week - 10

a list of activities with their starting time and finishing time goal is to select maximum number of activities that can be performed single person such that selected activities must be non conflicting. ctivity is said to be non - conflicting if starting time of an activity than or equal to the finishing time of the other activity. that a person can only work on a single activity at a time.

m:→

1) START
2) input n
3) i, s[n], f[n]
4) if i>=n goto step 7
5) input s[i]
6) if i<n goto step 4
7) if i=0
8) if i>=n goto step 11
9) input f[i]
10) if i<n goto step 8
11) vector <vector<int>> a, vector <int> act
12) if i>=n goto step 14
13) a.push_back ({f[i], (s[i], i+1)})
14) sort (a.begin(), a.end())
15) e = INT-MIN, c=0
16) if i>=n, goto step 19
17) if (a[i][1] >= e)
    (i) e = a[i][0]
    (ii) c++
    (iii) act.push_back (a[i][2])
18) if i<n goto step 16
19) print, "No. of non-conflicting ", c
20) print, "No. of activity selected"
21) if i>=n goto step
22) print act[i]
23) STOP

a long list of tasks. Each task takes specific time to accomplish and each task has a deadline associated with it. You have to design algorithm and implement it using a program to find maximum no. of tasks that can be completed without crossing their deadlines also find list of selected tasks.

Algorithm:→

1) START
2) input n
3) i, t[n], f[n]
4) if i>=n goto step 7
5) input t[i]
6) if i<n goto step 4
7) i=0
8) if i>n goto step 11
9) input f[i]
10) if i<n goto step 8
11) vector <vector <int>> a, vector <int> act
12) if i>=n goto step 14
13) a.push-back ({f[i], f[i]-t[i], i+1})
14) sort(a.begin(), a.end())
15) e = INT_MIN, c=0
16) if i>=n goto step 12
17) if (a[i][1]>=e)
18) e = a[i][0]
19) c++
20) act.push-back (a[i][2])
21) if i<n goto step 16
22) sort (act.begin(), act.end())
23) Print c
24) Print, "Selected Task Numbers"
25) if i>=n goto step 23
26) Print act[i]
27) if i<n goto step 25
28) STOP

an unsorted array of elements, design an algorithm and implem
ng a program to find whether majority element exists or not.
d find median of the array. A majority element is an element
ppears more than n/2 times, where n is the size of array.

m:→

1) START
2) input n
3) i, a[n], i, j
4) if i >= n goto step 6
5) input a[i]
6) f = 0
7) sort (a, a+n)
8) if i >= n goto step 14
9) c = 1
10) j = i+1
11) while(j<n && a[j++] == a[i])
    (i) c++
12) if(c > n/2) Print "Yes"
    i) f = 1
    ii) break
13) i = j-1
14) if(f == 0) Print "No"
15) if(n%2 != 0) Print a[n/2]
16) else Print a[n/2] + a[n/2-1]/2
17) STOP