GREEDY ALGORITHMS

QUES-1 **Activity Selection**

**Link-** [**Activity Selection**](https://practice.geeksforgeeks.org/problems/activity-selection/0)

Solution-

#include<bits/stdc++.h>

#include<iostream>

using namespace std;

int main()

{

int t,n;

int count=1;

map<int,int>mymap;

cin>>t;

while(t--)

{

cin>>n;

int a[n],b[n];

cout<<"Enter START TIME OF ALL ACTIVITIES: \n";

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

{

cin>>a[i];

}

cout<<"\nEnter END TIME OF ALL ACTIVITIES: \n";

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

{

cin>>b[i];

}

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

{

mymap.insert(make\_pair(b[i],a[i]));}

map<int,int>:: iterator i;

map<int,int>:: iterator j;

i=mymap.begin();

j=mymap.begin();

j++;

while(j!=mymap.end())

{

if(i->first<=j->second)

{ count++;i=j;}

j++;

}

cout<<"\nTOTAL ACTIVITIES :- "count;

count=0;

}

return 0;

}

QUES-2 **Coin Piles**

**Link-** [**Coin Piles**](https://practice.geeksforgeeks.org/problems/coin-piles/0)

Solution-

#include<bits/stdc++.h>

using namespace std;

int coinpiles(int[],int,int);

int main()

{

int n,k,i,t;

cin>>t;

while(t--){

cin>>n>>k;

int a[n];

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

cin>>a[i];

cout<<coinpiles(a,n,k)<<endl;

}

return 0;

}

int coinpiles(int a[],int n,int k)

{

int i=0,j=0,count=0,x,t=0,z=(n-1)-i;

sort(a,a+n,greater<int>());

while(i!=n)

{

j++;

if(a[i]-a[j]>k)

{

x=a[i]-a[n-1];

count=count+(x-k);

i++;

j=i;

z=(n-1)-i;

t=0;

}

else{t++;}

if(t==z)

{break;}

//cout<<"value of count = "<<count<<"t= "<<t<<"z= "<<z;

}

return count;}

QUES-3 **Maximize Toys**

**Link-** [**Maximize Toys**](https://practice.geeksforgeeks.org/problems/maximize-toys/0)

Solution-

#include<bits/stdc++.h>

using namespace std;

int maximizetoys(int [],int,int);

int main()

{

int t,n,i,k;

cin>>t;

while(t--)

{

cin>>n>>k;

int a[n];

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

{

cin>>a[i];

}

cout<<maximizetoys(a,n,k);

}

return 0;

}

int maximizetoys(int a[],int n,int k)

{

int i,sum=0;

sort(a,a+n);

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

{

sum=sum+a[i];

if(sum>k)

break;

}

return i;

}

QUES-4 **Largest number possible**

**Link-** [**Largest number possible**](https://practice.geeksforgeeks.org/problems/largest-number-possible/0)

//---------------------O(nlogn)-------------------

#include<bits/stdc++.h>

using namespace std;

int main()

{

int t;

cin>>t;

while(t--)

{

int n,i,sum=0;

cin>>n;

int a[n],b[n];

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

cin>>a[i];

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

cin>>b[i];

sort(a,a+n);

sort(b,b+n,greater<int>());

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

sum=sum+a[i]\*b[i];

cout<<sum;

}

return 0;

}

**Ques-5 Shop in Candy Store**

**Link-** [**Shop in Candy Store**](https://practice.geeksforgeeks.org/problems/shop-in-candy-store/0)

-----------------------o(nlogn)--------------------------------

#include<bits/stdc++.h>

using namespace std;

int main()

{

int t,n,k,i;

cin>>t;

while(t--)

{

int maxsum=0,minsum=0;

cin>>n>>k;

int a[n];

int x=n-k;

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

cin>>a[i];

sort(a,a+n);

for(i=0;i<x;i++)

{minsum=minsum+a[i];

maxsum=maxsum+a[n-i-1];

}

cout<<minsum<<" "<<maxsum<<endl;

}

return 0;

}

**Ques-6 Geek collects the balls**

**Link-**[**Geek collects the balls**](https://practice.geeksforgeeks.org/problems/geek-collects-the-balls/0)

**Sol1-Using Linked List**

#include<bits/stdc++.h>

**using** **namespace** std;

**struct** node1

{

int data;

**struct** node1\* next;

};

**struct** node2

{

int data;

**struct** node2\* next;

};

**struct** node1\* head1=NULL;

**struct** node2\* head2=NULL;

void createnode1(int data)

{

**struct** node1\* newnode1=**new** node1;

newnode1->data=data;

newnode1->next=head1;

head1=newnode1;

}

void createnode2(int data)

{

**struct** node2\* newnode2=**new** node2;

newnode2->data=data;

newnode2->next=head2;

head2=newnode2;

}

void display1()

{

**struct** node1\* temp=head1;

**while**(temp!=NULL)

{

cout<<temp->data<<" ";

temp=temp->next;

}

}

void display2()

{

**struct** node2\* temp=head2;

**while**(temp!=NULL)

{

cout<<temp->data<<" ";

temp=temp->next;

}}

void reverse1 ()

{

**struct** node1 \*c;

**struct** node1 \*p;

**struct** node1 \*n;

c = head1;

p = NULL;

**while** (c != NULL)

{

n = c->next;

c->next = p;

p = c;

c = n;

}

head1 = p;

}

void reverse2()

{

**struct** node2 \*c;

**struct** node2 \*p;

**struct** node2 \*n;

c = head2;

p = NULL;

**while** (c != NULL)

{

n = c->next;

c->next = p;

p = c;

c = n;

}

head2 = p;

}

int collectballs(**struct** node1\* cur1, **struct** node2\* cur2,char road)

{int temp,sum=0;

**while**(cur1!=NULL && cur2!=NULL)

{

**if**(cur1->data==cur2->data)

{

**if**(road=='A')

{

temp=cur1->data;

road='B';

cur1=cur1->next;

cur2=cur2->next;

}**else** **if**(road=='B')

{

temp=cur2->data;

road='A';

cur1=cur1->next;

cur2=cur2->next;

}

}

*//a>b*

**else** **if**(cur1->data>cur2->data)

{

**if**(road=='A')

{

temp=cur1->data;

cur1=cur1->next;

cur2=cur2->next;

}

**else** **if**(road=='B')

{

temp=cur2->data;

cur2=cur2->next;

}

}

**else** **if**(cur1->data<cur2->data)

{

**if**(road=='A')

{

temp=cur1->data;

cur1=cur1->next;

}

**else** **if**(road=='B')

{

temp=cur2->data;

cur1=cur1->next;

cur2=cur2->next;

}

}

sum=sum+temp;

}

*//if list size is not equal;*

**if**(cur1!=NULL)

{

**while**(cur1!=NULL)

{

sum=sum+cur1->data;

cur1=cur1->next;

}

}

**else** **if**(cur2!=NULL)

{

**while**(cur2!=NULL)

{

sum=sum+cur2->data;

cur2=cur2->next;

}

}

**return** sum;

}

int main()

{

char road;

int n1,n2,x,t;

cin>>t;

**while**(t--){

cin>>n1;

cin>>n2;

**for**(int i=0;i<n1;i++)

{cin>>x;createnode1(x);}

**for**(int i=0;i<n2;i++)

{cin>>x;createnode2(x);}

reverse1();

reverse2();

**if**(head1->data<head2->data)

{road='B';}

**else**

{road='A'; }

cout<<collectballs(head1,head2 ,road)<<endl;

}

**return** 0;}