1. GREEDY ALGORITHM PROGRAMS

1. Write a program to take value V and  we want to make change for V Rs, and we have infinite supply of each of the denominations in Indian currency, i.e., we have infinite supply of { 1, 2, 5, 10, 20, 50, 100, 500, 1000} valued coins/notes, what is the minimum number of coins and/or notes needed to make the change.

Input Format:

Take an integer from stdin.

Output Format:

print the integer which is change of the  number.

Example Input :

64

Output:

4

Explanaton:

We need a 50 Rs note and a 10 Rs note and two 2 rupee coins.

Program:

#include <stdio.h>

int main () {

int a ;

scanf("%d",&a);

int c=0;

int arr [] ={1000,500,100,50,20,10,5,2,1};

int nd=sizeof(arr)/sizeof(arr[0]);

for (int i=0;i<nd;i++){

if(a>=arr[i]){

a=a-arr[i];

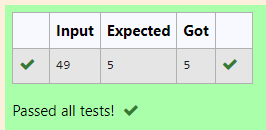
c++;

}

}

printf("%d",c);

}



2. Assume you are an awesome parent and want to give your children some cookies. But, you should give each child at most one cookie.

Each child i has a greed factor g[i], which is the minimum size of a cookie that the child will be content with; and each cookie j has a size s[j]. If s[j] >= g[i], we can assign the cookie j to the child i, and the child i will be content. Your goal is to maximize the number of your content children and output the maximum number.

**Example 1:**

**Input:**

3

1 2 3

2

1 1

**Output:**

1

Explanation: You have 3 children and 2 cookies. The greed factors of 3 children are 1, 2, 3.

And even though you have 2 cookies, since their size is both 1, you could only make the child whose greed factor is 1 content.

You need to output 1.

**Constraints:**

1 <= g.length <= 3 \* 10^4

0 <= s.length <= 3 \* 10^4

1 <= g[i], s[j] <= 2^31 - 1

PROGRAM:

#include<stdio.h>

int main (){

int a,b;

scanf("%d",&a);

int arr[a];

for (int i=0;i<a;i++){

scanf("%d",&arr[i]);

}

scanf("%d",&b);

int brr[b];

for (int i=0;i<b;i++){

scanf("%d",&brr[i]);

}

for (int i=a;i>0;i--){

for (int j=b;j>0;j--){

if (arr[i]==brr[i]){

printf("%d",arr[i]);

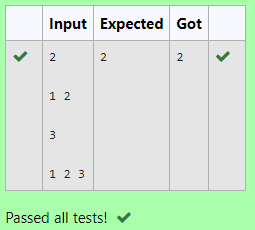
break;

}

}

}

}



A person needs to eat burgers. Each burger contains a count of calorie. After eating the burger, the person needs to run a distance to burn out his calories.   
 If he has eaten *i* burgers with c calories each, then he has to run at least *3i \* c*  kilometers to burn out the calories. For example, if he ate 3  
 burgers with the count of calorie in the order: [1, 3, 2], the kilometers he needs to run are (30 \* 1) + (31 \* 3) + (32 \* 2) = 1 + 9 + 18 = 28.  
 But this is not the minimum, so need to try out other orders of consumption and choose the minimum value. Determine the minimum distance  
 he needs to run. Note: He can eat burger in any order and use an efficient sorting algorithm.Apply greedy approach to solve the problem.  
**Input Format**First Line contains the number of burgers  
Second line contains calories of each burger which is n space-separate integers   
   
**Output Format**Print: Minimum number of kilometers needed to run to burn out the calories  
   
**Sample Input**   
3  
5 10 7  
**Sample Output**76

PROGRAM:

#include <stdio.h>

#include <math.h>

int main () {

int t ,k=0;

scanf("%d",&t);

int arr[t];

for (int i=0;i<t;i++){

scanf("%d",&arr[i]);

}

int temp=arr[0];

for (int i=0;i<t;i++){

for (int j=0;j<t;j++){

if (arr[i]>arr[j]){

temp=arr[i];

arr[i]=arr[j];

arr[j]=temp;

}

}

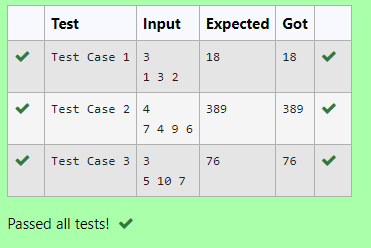
}

for (int i=0;i<t;i++){

k+=((pow(t,i))\*arr[i]);

}

printf("%d",k);

}

4. Given an array of N integer, we have to maximize the sum of arr[i] \* i, where i is the index of the element (i = 0, 1, 2, ..., N).Write an algorithm based on Greedy technique with a Complexity O(nlogn).

 Input Format:

First line specifies the number of elements-n

The next n lines contain the array elements.

Output Format:

Maximum Array Sum to be printed.

Sample Input:

5

2 5 3 4 0

Sample output:

40

PROGRAM: #include <stdio.h>

int main(){

int a ;

scanf("%d",&a);

int arr[a];

int sum=0;

for (int i=0;i<a;i++){

scanf("%d",&arr[i]);

}

int temp=arr[0];

for (int i=0;i<a;i++){

for (int j=0;j<a;j++){

if (arr[i]<arr[j]){

temp=arr[i];

arr[i]=arr[j];

arr[j]=temp;

}

}

}

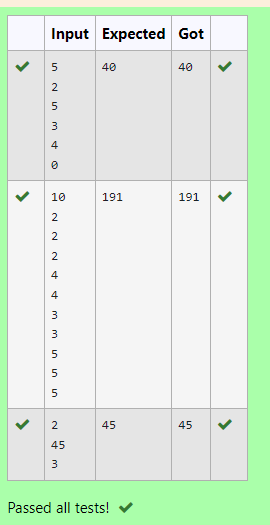
for (int i=0;i<a;i++){

sum+=(arr[i]\*i);

}

printf("%d",sum);

}



5. Given two arrays array\_One[] and array\_Two[] of same size N. We need to first rearrange the arrays such that the sum of the product of pairs( 1 element from each) is minimum. That is SUM (A[i] \* B[i]) for all i is minimum.

**For example:**

| **Input** | **Result** |
| --- | --- |
| 3  1  2  3  4  5  6 | 28 |

PROGRAM :

#include <stdio.h>

int main ( ){

int a;

scanf("%d",&a);

int arr[a];

int brr[a];

for (int i=0;i<a;i++){

scanf("%d",&arr[i]);

}

for (int i=0;i<a;i++){

scanf("%d",&brr[i]);

}

int temp;

for(int i=0;i<a;i++){

for (int j=0;j<a;j++){

if (arr[i]>arr[j]){

temp=arr[i];

arr[i]=arr[j];

arr[j]=temp;

}

}

}

for(int i=0;i<a;i++){

for (int j=0;j<a;j++){

if (brr[i]<brr[j]){

temp=brr[i];

brr[i]=brr[j];

brr[j]=temp;

}

}

}

int sum=0;

for (int i=0;i<a;i++){

sum+=arr[i]\*brr[i];

}

printf("%d",sum);

}

