

Given two numbers, write a C program to swap the given numbers.

For example:

Input	Result
10 20	20 10

Answer: (penalty regime: 0 %)

```
1 #include<stdio.h>
2 int main(){
3     int a;
4     int b;
5     scanf("%d %d",&a,&b);
6     printf("%d %d",b,a);
7     return 0;
8 }
```

	Input	Expected	Got	
✓	10 20	20 10	20 10	✓

Passed all tests! ✓

Correct

Marks in Maths >= 65
Marks in Physics >= 55
Marks in Chemistry >= 50
Or
Total in all three subjects >= 180

Sample Test Cases

Test Case 1

Input

70 80 80

Output

The candidate is eligible

Test Case 2

Input

50 80 80

Output

The candidate is eligible

Test Case 3

Input

50 60 40

Output

The candidate is not eligible

Answer: (usually ranges: 0-70)

```
1 //CandidateEligible.java
2
3 import java.util.Scanner;
4
5 public class CandidateEligible {
6     public static void main(String[] args) {
7         Scanner sc = new Scanner(System.in);
8         int Maths = sc.nextInt();
9         int Physics = sc.nextInt();
10        int Chemistry = sc.nextInt();
11
12        if (Maths >= 65 || Physics >= 55 || Chemistry >= 50 || Maths + Physics + Chemistry >= 180) {
13            System.out.println("The candidate is eligible");
14        } else {
15            System.out.println("The candidate is not eligible");
16        }
17    }
18 }
19
20 return 0;
```

Input	Expected	Got
70 80 80	The candidate is eligible	The candidate is eligible ✓
50 80 80	The candidate is eligible	The candidate is eligible ✓
50 60 40	The candidate is not eligible	The candidate is not eligible ✓

Passed all tests ✓

Correct
Marks for this submission: 100/100

Saba is very kind to beggars and every day Saba donates half of the amount he has when ever a beggar requests him. The money M left in Saba's hand is passed as the input and the number of beggars B who received the alms are passed as the input. The program must print the money Saba had in the beginning of the day.

Input Format:

The first line denotes the value of M.
The second line denotes the value of B.

Output Format:

The first line denotes the value of money with Saba in the beginning of the day.

Example Input/Output:

Input:

100
2

Output:

400

Explanation:

Saba donated to two beggars. So when he encountered second beggar he had $100*2 = Rs.200$ and when he encountered 1st he had $200*2 = Rs.400$.

Answer: (usually wrong: 0 %)

```
1 //SabaDonation.java
2 import java.util.Scanner;
3 public class SabaDonation {
4     public static void main(String[] args) {
5         Scanner sc = new Scanner(System.in);
6         int M = sc.nextInt();
7         int B = sc.nextInt();
8         while (B > 0) {
9             M = M * 2;
10             B--;
11         }
12         System.out.println(M);
13     }
14 }
```

	Input	Expected	Got	
✓	100	100	100	✓
	2			

Passed all tests. ✓

Connect

Marks for this submission: 7.00/1.00

Previous page

Next page

The CEO of company ABC, Inc. wanted to encourage the employees coming on time to the office. So he announced that for every consecutive day an employee comes on time in a week (starting from Monday to Saturday), he will be awarded \$x,200 more than the previous day as "Punctuality incentive". The incentive 1 for the starting day (ie on Monday) is passed as the input to the program. The number of days, N an employee came on time consecutively starting from Monday is also passed as the input. The program must calculate and print the "Punctuality incentive" P of the employee.

Input Format:

The first line denotes the value of x.
The second line denotes the value of N.

Output Format:

The first line denotes the value of P.

Example Input/Output:

Input:

500
3

Output:

2100

Explanation:

On Monday the employee receives \$x,500, on Tuesday \$x,700, on Wednesday \$x,900

So total = \$x,2100

Answer: (usually requires 3 N)

```
1 #include<stdio.h>
2 int main()
3 {
4     int A,x,count=0;
5     scanf("%d",&x);
6     for(int i=0;i<N;i++){
7         count+=count;
8         printf("%d",count);
9     }
10     return 0;
11 }
```

	Input	Expected	Got	
✓	100	2100	2100	✓
✓	x			
✓	100	900	900	✓
✓	x			

Percent of tests: ✓

Correct

Marks for this submission: 1.00/1.00

Previous page

Next page

Two numbers M and N are passed as the input. A number X is also passed as the input. The program must print the numbers divisible by X from N to M (inclusive of M and N).

Input Format:

The first line denotes the value of M.
The second line denotes the value of N.
The third line denotes the value of X.

Output Format:

Numbers divisible by X from N to M, with each number separated by a space.

Boundary Conditions:

1 <= M <= 9999999
M < N <= 9999999
1 <= X <= 9999

Example Input/Output 1:

Input:
2
40
7

Output:
35 28 21 14 7

Example Input/Output 2:

Input:
66
121
11

Output:
121 110 99 88 77 66

Answer: (yourly regime: 0%)

```
1 #include<iostream.h>
2 int main()
3 {
4     int M;
5     int N;
6     scanf("%d %d",&M,&N,&X);
7     for (int i=N;i>=M;i--)
8     {
9         if(i%X==0){
10             printf("%d ",i);
11         }
12     }
13 }
```

	Input	Expected	Got	
✓	2	15 20 25 30 2	15 20 25 30 2	✓
✗	7			

Passed all tests! ✓

Comment

Works for 10 submissions (100%)

Two numbers M and N are passed as the input. A number X is also passed as the input. The program must print the numbers divisible by X from N to M (inclusive of M and N).

Input Format:

The first line denotes the value of M
The second line denotes the value of N
The third line denotes the value of X

Output Format:

Numbers divisible by X from N to M, with each number separated by a space.

Boundary Conditions:

$$\begin{aligned} I &\subset M \subset 999999 \\ M &\subset N \subset 999999 \\ I &\subset X \subset 9999 \end{aligned}$$

Example Input/Output 1:

input:
2
40
7

Output:
35 28 21 14 7

Example Input/Output 2:

```
Input:
60
120
11
```

Output:
121 110 99 88 77 66

Answer: (personality) requires 0/90

```

1 //main.c:ordina.h
2 int main(){
3     int N;
4     int M;
5     int K;
6     scanf("%d %d %d", &N, &M, &K);
7     for (int i=0; i<M; i++){
8         if (i%K==0){
9             printf("%d ", i);
10        }
11    }
12    return 0;
13 }

```

	Input	Expected	Got	
✓	0 10 2	15 20 25 35 /	15 20 25 35 /	✓

Passed all tests! ✓

Correct

Wages for this submission: 1,000.00

Write a C program to find the quotient and remainder of given integers.

For example:

Input	Result
12	1
6	0

Answer: (partialy right: 0 %)

```
1 // quotient.c
2 #include <stdio.h>
3 int main()
4 {
5     int a,b;
6     scanf("%d %d",&a,&b);
7     int c=a/b;
8     int d=a%b;
9     printf("Quotient: %d, Remainder: %d",c,d);
10    return 0;
11 }
```

	Input	Expected	Got	
✓	12	1	1	✓
	6	0	0	

Passed all testcases ✓

Correct

Marks for this submission: 1.00/1.00.

Write a C program to find the biggest among the given 3 integers?

For example:

Input	Result
10 20 30	30

Answer: (usually requires 15 Nk)

```
1 #include<stdio.h>
2 int main()
3 {
4     int a,b,c;
5     scanf("%d %d %d",&a,&b,&c);
6     if(a>b & a>c){
7         printf("%d\n",a);
8     }
9     else if (b>a & b>c){
10        printf("%d\n",b);
11    }
12    else{
13        printf("%d\n",c);
14    }
15    return 0;
16 }
17
```

	Input	Expected	Got
✓	10 20 30	30	✓

Passed all tests! ✓

Correct

Thanks for the submission: 1.10/1.00

Write a C program to find whether the given integer is odd or even?

For example:

Input	Result
12	Even
11	Odd

Answer: (penalty regime: 0 %)

```
1 //odd-even.c
2 int main()
3 {
4     int n;
5     scanf("%d",&n);
6     if(n%2 == 0)
7     {
8         printf("Even");
9     }
10    else
11    {
12        printf("Odd");
13    }
14 }
```

	Input	Expected	Got
✓	12	Even	Even ✓
✓	11	Odd	Odd ✓

Passed all tests ✓

Correct

Stars for this submission: 1,004/1,000

Write a C program to find the factorial of given n.

For example:

Input	Result
5	120

Answer: (usually requires O(N))

```
1 #include<stdio.h>
2 int main()
3 {
4     int fact = 1;
5     scanf("%d",&n);
6     for (int i=1;i<=n;i++){
7         fact *=i;
8     }
9     printf("%d",fact);
10 }
```

	Input	Expected	Got
✓	5	120	120 ✓

Passed all testcases ✓

Correct

Marked for this submission: 1.00/1.00

Write a C program to find the sum first N natural numbers.

For example:

Input	Result
5	15

Answer: (penalty regime: 0 %)

```
1 #include<stdio.h>
2 int main()
3 {
4     int n,sum=0;
5     scanf("%d",&n);
6     for(int i=1;i<=n;i++)
7     {
8         sum+=i;
9     }
10    printf("%d",sum);
11 }
```

	Input	Expected	Got
✓	5	15	15 ✓

Passed all tests ✓

Correct

Marked for this submission: 1.00/1.00

Write a C program to find the nth term in the Fibonacci series.

For example:

Input	Result
0	0
1	1
2	1

Answer: (partialy rigine: 0%)

```
1 #include <stdio.h>
2 int main()
3 {
4     int n,a=0,b=1,c;
5     scanf("%d",&n);
6     if(n==0)
7     {
8         printf("0");
9     }
10    else if(n==1)
11    {
12        printf("1");
13    }
14    else
15    {
16        for(int i=2;i<=n;i++)
17        {
18            c=a+b;
19            a=b;
20            b=c;
21        }
22        printf("%d", b);
23    }
24 }
```

	Input	Expected	Got	
✓	0	0	0	✓
✓	1	1	1	✓
✓	2	1	1	✓

Passed all tests: ✓

Correct

Score for this submission: 1.00/1.00

[Previous page](#)

[Next page](#)

Write a C program to find the power of integers.

input:

a b

output:

a^b value

For example:

Input	Result
2 3	8

Answer: (usually requires 0 %)

```
1 #include<stdio.h>
2 #include<math.h>
3 int main()
4 {
5     int a,b;
6     scanf("%d %d",&a,&b);
7     printf("%d",(int)pow(a,b));
8 }
```

	Input	Expected	Got
✓	2 3	8	8 ✓

Passed all tests ✓

Correct

Marks for this submission: 1.00/1.00

Write a C program to find Whether the given integer is prime or not.

For example:

Input	Result
7	Prime
8	No Prime

Answer: (penalty: 0 %)

```
1 #include<stdio.h>
2 int main()
3 {
4     int n,flag=0;
5     scanf("%d",&n);
6
7     if (n==0 || n==1)
8     {
9         flag=1;
10    }
11    else{
12        for(int i = 2; i <= n / 2; i++)
13        {
14            if(n%i==0){
15                flag=1;
16                break;
17            }
18        }
19    }
20    if(flag==0){
21        printf("Prime");
22    }
23    else{
24        printf("No Prime");
25    }
26 }
```

	Input	Expected	Got	
✓	7	Prime	Prime	✓
✓	8	No Prime	No Prime	✓

Passed all tests ✓

Correct

Marks for this submission: 1.00/1.00

[Previous page](#)

[Next page](#)

Write a C program to find the reverse of the given integer?

Answer: (usually require: 0 %)

```
1 //reverse.c
2 int main()
3 {
4     int a,reversed=0,remainder;
5     scanf("%d",&a);
6     while(a!=0)
7     {
8         remainder=a%10;
9         reversed=reversed*10+remainder;
10        a/=10;
11    }
12    printf("%d\n",reversed);
13 }
```

	Input	Expected	Got
✓	123	321	321 ✓

Passed all tests! ✓

Correct

Marks for this submission: 1.00/1.00.

Question 1 | Correct Mark 1.00 out of 1.00 Flag question

Convert the following algorithm into a program and find its time complexity using the counter method.
void function (int x)

```
{
    int i = 1;

    int k = 1;

    while(x <= i)
    {
        i++;
        k = k * i;
    }
}
```

Note: No need of counter increment for Declaration and scanf() and count variable printf() statements.

Input:

A positive integer x

Output:

Print the value of the counter variable

For example:

Input	Result
8	12

Answer: (space complexity: O 1)

```
1 #include <stdio.h>
2
3 void function (int n)
4 {
5     int counter = 0;
6     int i = 1;
7     int k = 1;
8     counter = 2;
9
10    while(x <= i)
11    {
12        counter++;
13        i++;
14        counter++;
15        k = i;
16        counter++;
17    }
18    counter++;
19    printf("%d\n", counter);
20 }
21
22 int main()
23 {
24     int n;
25     scanf("%d", &n);
26     function(n);
27     return 0;
28 }
```

	Input	Expected	Got
✓	8	12	12 ✓
✓	4	8	8 ✓

Passed all tests! ✓

Correct

Marks for this submission: 1.00/1.00.

Convert the following algorithm into a program and find its time complexity using the counter method.

void func(int n)

```
{
    if(n==1)
    {
        printf("%d");
    }
    else
    {
        for(int i=1; i<=n; i++)
        {
            for(int j=1; j<=i; j++)
            {
                printf("%d");
                printf("%d");
                break;
            }
        }
    }
}
```

Note: No need of counter increment for declaration and scanf() and count variable printf() statements.

Input:

A positive integer n

Output:

Print the value of the counter variable

Answer: (usually ranges: 0 %)

```
1 #include <stdio.h>
2
3 void func(int n) {
4     long long counter = 0;
5
6     if (n == 1) {
7         counter += 2;
8     } else {
9         counter = 511 * n + 2;
10    }
11    printf("%lld\n", counter);
12 }
13
14
15 int main() {
16     int n;
17     scanf("%d", &n);
18     func(n);
19     return 0;
20 }
```

	Input	Expected	Got	
✓	2	52	52	✓
✓	1000	51002	51002	✓
✓	100	712	712	✓

Passed all test! ✓

Correct

Marked for this submission: 1.00/1.00

Convert the following algorithm into a program and find its time complexity using counter method.

```
factor(num) {  
    {  
        for (i = 1; i <= num; ++i)  
        {  
            if (num % i == 0)  
            {  
                printf("%d ", i);  
            }  
        }  
    }  
}
```

Note: No need of counter increment for declarations and scanf() and counter variable printf() statement.

Input:

A positive integer n

Output:

Print the value of the counter variable

Answer:

```
1 #include <stdio.h>  
2  
3 int main() {  
4     int num, c = 0;  
5     scanf("%d", &num);  
6  
7     for (int i = 1; i <= num; ++i) {  
8         c++;  
9         if (num % i == 0) {  
10            c++;  
11        }  
12    }  
13  
14    printf("%d", c);  
15    return 0;  
16 }  
17  
18  
19  
20
```

	Input	Expected	Got	
✓	12	45	45	✓
✓	25	40	40	✓
✓	4	12	12	✓

Passed all tests! ✓

Correct

Marks for this submission: 1.00/1.00.

Convert the following algorithm into a program and find its time

complexity using counter method.

```
void function(int n)
{
    int c = 0;
    for(int i=n/2; i<n; i++)
        for(int j=i; j<n; j = 2 * j)
            for(int k=i; k<n; k = k * 2)
                c++;
}
```

Note: No need of counter increment for declarations and scanf() and count variable printf() statements.

Input:

A positive integer n

Output:

Print the value of the counter variable

Answer:

```
1 #include <stdio.h>
2
3 int main() {
4     int n, count = 0;
5     scanf("%d", &n);
6     int c = 0;
7     count++;
8
9     for (int i = n / 2; i < n; i++) {
10         count++;
11         for (int j = i; j < n; j = 2 * j) {
12             count++;
13             for (int k = i; k < n; k = k * 2) {
14                 count++;
15                 c++;
16                 count++;
17             }
18             count++;
19         }
20         count++;
21     }
22     count++;
23     printf("%d", count);
24     return 0;
25 }
```

	Input	Expected	Got	
✓	5	36	36	✓
✓	10	212	212	✓

Passed all tests! ✓

Correct

Marked for this submission: 1.00/1.00.

Convert the following algorithm into a program and find its time complexity using counter method.

```
void reverse(int n)
{
    int rev = 0, remainder;
    while (n != 0)
    {
        remainder = n % 10;
        rev = rev * 10 + remainder;
        n /= 10;
    }
    printf("%d", rev);
}
```

Note: No need of counter increment for declaration and scanf() and count variable printf() statements.

Input:

A positive integer n

Output:

Print the value of the counter variable

Answer:

```
1 #include<stdio.h>
2
3 int main() {
4     int n,c=0;
5     scanf("%d",&n);
6     int rev=0,remainder; c++;
7     while(n!=0) { c++;
8         remainder=n%10; c++;
9         rev=rev*10+remainder; c++;
10        n/=10; c++;
11    } c++;
12    c++;
13    printf("%d",c);
14    return 0;
15 }
16
```

	Input	Expected	Got	
✓	12	11	11	✓
✓	12.01	19	19	✓

Passed all tests! ✓

Correct

Marks for this submission: 1.00/1.00.

Given an array of 1s and 0s, find out if 1s are followed by 0s. Write a program using Divide and Conquer to Count the number of ones in the given array.

West Coast Fisheries, Inc. v. *State of Oregon* 1000

Charged Account

How many children sleep in the same room as the parent?

```

1 // Randomly select two
2
3 // swap them if not sorted
4 int swap(int a, int b) {
5     int temp = a;
6     a = b;
7     b = temp;
8 }
9
10 // compare two elements
11 bool compare(int a, int b) {
12     return a < b;
13 }
14
15 // partition the array
16 int partition(int arr[], int low, int high) {
17     int pivot = arr[low];
18     int i = low + 1;
19     int j = high;
20     while (i <= j) {
21         while (i <= j && compare(arr[i], pivot))
22             i++;
23         while (j > i && !compare(arr[j], pivot))
24             j--;
25         if (i < j)
26             swap(arr[i], arr[j]);
27     }
28     swap(arr[low], arr[i]);
29     return i;
30 }
31
32 // recursive function to sort the array
33 void quickSort(int arr[], int low, int high) {
34     if (low < high) {
35         int pi = partition(arr, low, high);
36         quickSort(arr, low, pi - 1);
37         quickSort(arr, pi + 1, high);
38     }
39 }
40
41 // Driver code
42 int main() {
43     int arr[] = {10, 20, 30, 40, 50, 60, 70, 80, 90, 100};
44     int n = sizeof(arr) / sizeof(arr[0]);
45     quickSort(arr, 0, n - 1);
46     for (int i = 0; i < n; i++)
47         cout << arr[i] << " ";
48     return 0;
49 }

```

It should be noted that the above results are based on the assumption that the data are stationary. If the data are non-stationary, the results may be biased. Therefore, it is important to test for stationarity before conducting the regression analysis.

12 months for 100% satisfaction. 7,857,385.

Given an array `nums` of size `n`, return the majority element.

The majority element is the element that appears more than $\lfloor n / 2 \rfloor$ times. You may assume that the majority element always exists in the array.

Example 1:

Input: `nums = [1,1,1]`

Output: `1`

Example 2:

Input: `nums = [2,2,1,1,1,2]`

Output: `2`

Constraints:

- $n == \text{nums.length}$
- $1 \leq n \leq 5 \times 10^4$
- $-2^{31} \leq \text{nums}[i] \leq 2^{31} - 1$

For example:

Input	Result
1 1 1 1	1
2 2 2 1 1 1 2	2

Answer: (yourly rejects: 0 %)

```
1 //leetcode.cpp
2
3 int majorityElement(int nums[], int n) {
4     int count = 0, candidate = 0;
5
6     for (int i = 0; i < n; i++) {
7         if (count == 0) {
8             candidate = nums[i];
9             count = 1;
10        }
11        else if (nums[i] == candidate) {
12            count++;
13        }
14        else {
15            count--;
16        }
17    }
18    return candidate;
19 }
20
21 int main() {
22     int n;
23     scanf("%d", &n);
24     int nums[n];
25
26     for (int i = 0; i < n; i++) {
27         scanf("%d", &nums[i]);
28     }
29
30     int result = majorityElement(nums, n);
31     printf("%d\n", result);
32
33     return 0;
34 }
```

	Input	Expected	Got
✓	1 1 1 1	1	1 ✓

Passed all tests! ✓

Correct

Marks for this submission: 1.00/1.00

Problem Statement:

Given a sorted array and a value x , the floor of x is the largest element in array smaller than or equal to x . Write divide and conquer algorithm to find floor of x .

Input Format

First Line Contains Integer n - Size of array

Next n lines Contains n numbers - Elements of an array

Last Line Contains Integer x - Value for x

Output Format

First Line Contains Integer - floor value for x

Answer: (generally requires $O(\log)$)

```
1 //include <stdio.h>
2 int find_floor(int arr[], int low, int high, int x, int floor) {
3     if (low > high) {
4         return floor;
5     }
6     int mid = (low + high) / 2;
7     if (arr[mid] == x) {
8         return arr[mid];
9     }
10    else if (arr[mid] > x) {
11        return find_floor(arr, low, mid - 1, x, floor);
12    }
13    else {
14        return find_floor(arr, mid + 1, high, x, arr[mid]);
15    }
16 }
17
18 int main() {
19     int n;
20     scanf("%d", &n);
21     int arr[n];
22     for(int i = 0; i < n; i++) {
23         scanf("%d", &arr[i]);
24     }
25     int x;
26     scanf("%d", &x);
27     int floor_value = find_floor(arr, 0, n - 1, x, -1);
28     if (floor_value == -1) {
29         printf("No floor found\n");
30     } else {
31         printf("%d\n", floor_value);
32     }
33     return 0;
34 }
```

	Input	Expected	Got	
✓	6	1	1	✓
	1			
	2			
	6			
	10			
	12			
	18			
	2			
✓	5	95	95	✓
	10			
	20			
	60			
	100			
	129			
	200			
✓	7	9	9	✓
	1			
	5			
	7			
	9			
	10			
	14			
	20			
	30			

Passed all tests! ✓

Correct

Stars for this submission: 1.00/1.00.

Problem Statement:

Given a sorted array of integers say arr[] and a number x. Write a recursive program using divide and conquer strategy to check if there exist two elements in the array whose sum = x. If there exist such two elements then return the numbers, otherwise print as "No".

Note: Write a Divide and Conquer Solution

Input Format

First Line Contains Integer n - Size of array.

Next n lines Contains n numbers - Elements of an array.

Last Line Contains Integer x - Sum Value

Output Format

First Line Contains Integer - Element1

Second Line Contains Integer - Element2 (Element 1 and Elements 2 together sums to value "x")

Answer: (complexity regime: O(N))

```
1 #include <stdio.h>
2
3 void find_pair(int arr[], int low, int high, int x) {
4     if (low >= high) {
5         printf("No\n");
6         return;
7     }
8
9     int sum = arr[low] + arr[high];
10
11     if (sum == x) {
12         printf("%d\n%d\n", arr[low], arr[high]);
13         return;
14     }
15     else if (sum < x) {
16         find_pair(arr, low + 1, high, x);
17     }
18     else {
19         find_pair(arr, low, high - 1, x);
20     }
21 }
22
23 int main() {
24     int n;
25     scanf("%d", &n);
26
27     int arr[n];
28     for (int i = 0; i < n; i++) {
29         scanf("%d", &arr[i]);
30     }
31
32     int x;
33     scanf("%d", &x);
34
35     find_pair(arr, 0, n - 1, x);
36
37     return 0;
38 }
```

	Input	Expected	Got	
✓	1 2 3 4 5	1 10 	1 10 	✓
✓	1 2 3 4 5 6 7 8 9	No 	No 	✓

Passed all tests! ✓

Correct

Marks for this submission: 1.00/1.00.

Write a Program to implement the Quick Sort Algorithm.

Input Format:

The first line contains the no. of elements in the list.

The next n lines contain the elements.

Output:

Sorted list of elements

For example:

Input	Result
5 67 87 52 98 76	52 67 76 87 98

Answer:

```

1 #include <stdio.h>
2 void swap(int *a, int *b) {
3     int temp = *a;
4     *a = *b;
5     *b = temp;
6 }
7 int partition(int arr[], int low, int high) {
8     int pivot = arr[high];
9     int i = low - 1;
10    for (int j = low; j < high; j++) {
11        if (arr[j] < pivot) {
12            i++;
13            swap(&arr[i], &arr[j]);
14        }
15    }
16    swap(&arr[i + 1], &arr[high]);
17    return i + 1;
18 }
19 void quickSort(int arr[], int low, int high) {
20    if (low < high) {
21        int pi = partition(arr, low, high);
22        quickSort(arr, low, pi - 1);
23        quickSort(arr, pi + 1, high);
24    }
25 }
26
27 int main() {
28    int n;
29    scanf("%d", &n);
30    int arr[n];
31    for (int i = 0; i < n; i++) {
32        scanf("%d", &arr[i]);
33    }
34    quickSort(arr, 0, n - 1);
35    for (int i = 0; i < n; i++) {
36        printf("%d ", arr[i]);
37    }
38    printf("\n");
39    return 0;
40 }

```

	Input	Expected	Got	
✓	5 67 87 52 98 76	52 67 76 87 98	52 67 76 87 98	✓
✓	10 5 56 78 90 42 54 55 58 99 337	5 54 55 42 56 58 78 90 99 337	5 54 55 42 56 58 78 90 99 337	✓
✓	12 9 8 7 6 5 4 3 2 1 10 11 99	1 2 3 4 5 6 7 8 9 10 11 99	1 2 3 4 5 6 7 8 9 10 11 99	✓

Passed all tests! ✓

Correct

Marks for this submission: 1.00/1.00

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

Explanations

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

Answer: (penalty regime: 0 %)

```
1 #include <stdio.h>
2
3 int main() {
4     int V;
5     scanf("%d", &V);
6
7     int denominations[] = {1000, 500, 100, 50, 20, 10, 5, 2, 1};
8     int count = 0;
9
10    for (int i = 0; i < 9; i++) {
11        while (V >= denominations[i]) {
12            V -= denominations[i];
13            count++;
14        }
15    }
16
17    printf("%d\n", count);
18    return 0;
19 }
20
```

	Input	Expected	Got
✓	64	4	4 ✓

Passed all tests! ✓

Correct

Marks for this submission: (10/1.05)

[Finish review](#)

Assume you are an expensive 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 $s[j] \geq g[i]$, we can assign the cookie j to the child i , and the child will be content. Your goal is to maximize the number of your content children and output the maximum number.

Example 1:

Input:

g

$[1, 2, 3]$

s

$[1, 1]$

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 sizes both 1 , you could only make the child whose greed factor is 1 content.

You need to output 1 .

Constraints:

$1 \leq \text{length}(g) \leq 10^4$

$1 \leq \text{length}(s) \leq 10^4$

$1 \leq g[i], s[j] \leq 10^5$

Answer: (strictly regular O(N))

```
1 #include <vector>
2 #include <algorithm>
3
4 int compare(const int &a, const int &b) {
5     return a < b;
6 }
7
8 int main() {
9     int n;
10    cin >> n;
11    vector<int> g(n);
12    vector<int> s(n);
13    for (int i = 0; i < n; i++) {
14        cin >> g[i];
15    }
16    for (int i = 0; i < n; i++) {
17        cin >> s[i];
18    }
19    sort(g.begin(), g.end());
20    sort(s.begin(), s.end());
21    int i = 0;
22    int j = 0;
23    int count = 0;
24    while (i < g.size() && j < s.size()) {
25        if (s[j] < g[i]) {
26            j++;
27        } else {
28            count++;
29            i++;
30            j++;
31        }
32    }
33    cout << count << endl;
34    return 0;
35 }
```

	Input	Expected	Got
✓	1 2 3	1	✓
	1 2		
	5		
	1 2 3		

Passed all test cases ✓

Submit

Wrong Answer (Submission #100100)

A person needs to eat burgers. Each burger contains a count of calories. 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 $x^2 + c$ kilometers to burn out the calories. For example, if he ate 2 burgers with the count of calories in the order: [1, 4, 2], the kilometers he needs to run are $(x^2 + 1) + (x^2 + 2) + 1 + x + 1x + 2x$. 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: We can not sort 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-separated integers.

Output Format:

Print: Minimum number of kilometers needed to run to burn out the calories.

Sample Input:

```
2
1 10 2
```

Sample Output:

```
76
```

For example:

Test	Input	Result
Test Case 1	1 1 10 2	18

Answer: (penalty regime: 0%)

```
1 #include <stdio.h>
2 #include <stdlib.h>
3 #include <math.h>
4
5 int cmp(const void *a, const void *b)
6 {
7     return (*(int*)b - *(int*)a);
8 }
9
10 int main()
11 {
12     int n;
13     scanf("%d", &n);
14     int a[n];
15     for(int i=0; i<n; i++)
16         scanf("%d", &a[i]);
17     qsort(a, n, sizeof(int), cmp);
18     int md=0;
19     for(int i=0; i<n; i++)
20         md+=a[i]*a[i];
21     printf("%d", md);
22 }
23
```

	Test	Input	Expected	Got	
✓	Test Case 1	1 1 10 2	18	18	✓
✓	Test Case 2	5 7 1 8 6	189	189	✓
✓	Test Case 3	2 1 10 2	76	76	✓

Passed all tests! ✓

Correct

Wrote for this submission: 1.00/1.00

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

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

Answer: (optimally runtime: 0.98)

```
1 #include <stdio.h>
2
3 int main() {
4     int n, sum = 0;
5     scanf("%d", &n);
6     int arr[n];
7     for (int i = 0; i < n; i++) {
8         scanf("%d", &arr[i]);
9     }
10    for (int i = 0; i < n - 1; i++) {
11        for (int j = 0; j < n - 1; j++) {
12            if (arr[j] > arr[j + 1]) {
13                int temp = arr[j];
14                arr[j] = arr[j + 1];
15                arr[j + 1] = temp;
16            }
17        }
18    }
19    for (int i = 0; i < n; i++) {
20        sum += arr[i] * i;
21    }
22    printf("%d", sum);
23 }
24
```

Input	Expected	Got	
5 2 5 3 4 0	40	40	✓
10 2 2 2 2 2 2 2 2 2	101	101	✓
3 15 2	15	15	✓

Passed all tests! ✓

Correct

Marks for this submission: 1.00/1.00.

Given two arrays `array1` and `array2` of same size `N`. We need to first rearrange the arrays such that the sum of the product of pairs `i` element from each is minimum. That is $\text{SUM}(a[i] * b[i])$ for all `i` is minimum.

For example:

Input	Result
1	20
1	
2	
4	
5	
6	

Answer: Optimaly regime: $O(N^2)$

```
1 #include<iostream>
2 #include<vector>
3 int compareVec(const std::vector<int>& a, const std::vector<int>& b) {
4     return *(int *)a - *(int *)b;
5 }
6 int compareVec(const std::vector<int>& a, const std::vector<int>& b) {
7     return *(int *)b - *(int *)a;
8 }
9 int minProductSum(const std::vector<int>& arr1, const std::vector<int>& arr2, int n) {
10     sort(arr1.begin(), arr1.end(), compareVec);
11     sort(arr2.begin(), arr2.end(), compareVec);
12     int sum = 0;
13     for (int i = 0; i < n; i++) {
14         sum += arr1[i] * arr2[i];
15     }
16     return sum;
17 }
18 int main() {
19     int n;
20     std::vector<int> arr1;
21     std::vector<int> arr2;
22     for (int i = 0; i < n; i++) {
23         int x;
24         std::cin >> x;
25         arr1.push_back(x);
26     }
27     for (int i = 0; i < n; i++) {
28         int x;
29         std::cin >> x;
30         arr2.push_back(x);
31     }
32     int n = arr1.size() / 2;
33     int result = minProductSum(arr1, arr2, n);
34     std::cout << result;
35 }
```

Input	Expected	Got
1 1 2 4 5 6	20	20
4 2 5 2 4 5	22	22
5 20 10 10 10 10 8 10 5 5 10	200	200

Passed all tests: ✓

Correct

Works for this submission: 1.00/1.00

Playing with Numbers:

Ram and Sita are playing with numbers by giving puzzles to each other. Now it was Ram's turn, so he gave Sita a positive integer 'n' and two numbers 1 and 3. He asked her to find the possible ways by which the number n can be represented using 1 and 3. Write any efficient algorithm to find the possible ways.

Example 1:

Input: 6

Output: 6

Explanation: There are 6 ways to 6 represent number with 1 and 3.

1+1+1+1+1+1
3+3
1+1+1+3
1+3+1+1
1+3+1+1
3+1+1+1

Input Format

First Line contains the number n

Output Format

Print: The number of possible ways 'n' can be represented using 1 and 3

Sample Input

6

Sample Output

6

Answer: (penalty begins: 0 %)

```
1 #include <stdio.h>
2
3 int main() {
4     int n;
5     scanf("%d", &n);
6
7     long long dp[n + 1];
8     for (int i = 0; i <= n; i++) {
9         dp[i] = 0;
10    }
11
12    dp[0] = 1;
13
14    for (int i = 1; i <= n; i++) {
15        dp[i] += dp[i - 1];
16        if (i >= 3) {
17            dp[i] += dp[i - 3];
18        }
19    }
20
21    printf("%lld\n", dp[n]);
22    return 0;
23 }
```

	Input	Expected	Got	
✓	6	6	6	✓
✓	25	8973	8973	✓
✓	100	2748281994721824	2748281994721824	✓

Passed all tests! ✓

Correct

Mark for this submission: 10.00/10.00

Playing with Chessboard:

Item is given with an $n \times m$ chessboard with each cell with a monetary value. Item stands at the (0,0), that is the position of the top left white rook. He is given a task to reach the bottom right black rook position ($n-1, m-1$) constrained that he needs to reach the position by traveling the maximum monetary path under the condition that he can only travel one step right or one step down the board. Help him to achieve it by providing an efficient DP algorithm.

Example:

Input:

3

1 2 4

2 3 4

8 7 1

Output:

19

Explanation:

Totally there will be 6 paths among that the optimal is

Optimal path value: $1+2+8+7+1=19$

Input Format

First line contains the integer n

The next n lines contain the $n \times m$ chessboard values

Output Format

Print Maximum monetary value of the path

Answer: (usually requires 0 %)

```
1 #include <vector>
2
3 int max(int a, int b) {
4     return a > b ? a : b;
5 }
6
7 int main() {
8     int n;
9     scanf("%d", &n);
10
11     int board[n][n];
12     int dp[n][n];
13
14     for (int i = 0; i < n; i++) {
15         for (int j = 0; j < n; j++) {
16             scanf("%d", &board[i][j]);
17         }
18     }
19
20     dp[0][0] = board[0][0];
21
22     for (int i = 1; i < n; i++) {
23         dp[i][0] = dp[i-1][0] + board[i][0];
24     }
25
26     for (int i = 1; i < n; i++) {
27         dp[i][i] = dp[i-1][i] + board[i][i];
28     }
29
30     for (int i = 1; i < n; i++) {
31         for (int j = 1; j < n; j++) {
32             dp[i][j] = max(dp[i-1][j], dp[i][j-1]) + board[i][j];
33         }
34     }
35
36     printf("%d\n", dp[n-1][n-1]);
37
38     return 0;
39 }
```

Input	Expected	Got	
3 1 2 4 2 3 4 8 7 1	19	19	✓
4 1 2 4 1 1 3 4 1 1 4 1 1 1 2 3	13	12	✓
4 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1	16	16	✓

Passed all testcases ✓

Correct

Thanks for this submission: 55.00/10.00

Given two strings find the length of the common longest subsequence (need not be contiguous) between the two.

Example:

s1: ggtab

s2: gtaxb

s1 a g g t a b

s2 g a t x a y b

The length is 4

Solving it using Dynamic Programming

For example:

Input	Result
aab	2
aab	

Answer: (penalty regime: 0 %)

```
1 #include <stdio.h>
2 #include <string.h>
3
4 int max(int a, int b) {
5     return a > b ? a : b;
6 }
7
8 int main() {
9     char a1[1001], a2[1001];
10    scanf("%s", a1);
11    scanf("%s", a2);
12
13    int n = strlen(a1);
14    int m = strlen(a2);
15
16    int dp[n + 1][m + 1];
17
18    for (int i = 0; i <= m; i++) {
19        for (int j = 0; j <= n; j++) {
20            if (i == 0 || j == 0)
21                dp[i][j] = 0;
22            else if (a1[i - 1] == a2[j - 1])
23                dp[i][j] = dp[i - 1][j - 1] + 1;
24            else
25                dp[i][j] = max(dp[i - 1][j], dp[i][j - 1]);
26        }
27    }
28
29    printf("%d\n", dp[n][m]);
30    return 0;
31 }
32
```

	Input	Expected	Got	
✓	aab	2	2	✓
✓	aab			
✓	aaab	3	3	✓
	aaab			

Passed all test! ✓

Correct

Marked for IPVA submission: 1.00/1.00

Problem statement:

Find the length of the Longest Non-decreasing Subsequence in a given Sequence.

eg:

Input:9

Sequence:-1,3,4,5,2,2,2,2,3]

the subsequence is [-1,2,2,2,2,3]

Output:6

Answer: (penalty regime: 0 %)

```
1 #include <stdio.h>
2
3 int max(int a, int b) {
4     return a > b ? a : b;
5 }
6
7 int main() {
8     int n;
9     scanf("%d", &n);
10
11     int A[n];
12     for (int i = 0; i < n; i++) {
13         scanf("%d", &A[i]);
14     }
15
16     int dp[n];
17     for (int i = 0; i < n; i++) {
18         dp[i] = 1;
19     }
20
21     for (int i = 1; i < n; i++) {
22         for (int j = 0; j < i; j++) {
23             if (A[i] >= A[j]) {
24                 dp[i] = max(dp[i], dp[j] + 1);
25             }
26         }
27     }
28
29     int maxlen = 0;
30     for (int i = 0; i < n; i++) {
31         if (dp[i] > maxlen) {
32             maxlen = dp[i];
33         }
34     }
35
36     printf("%d\n", maxlen);
37     return 0;
38 }
```

	Input	Expected	Got	
✓	9 -1 3 4 5 2 2 2 2 3	6	6	✓
✓	7 3 2 2 1 4 7 6	5	5	✓

Passed all tests! ✓

Correct

Marks for this submission: 1.00/1.00.

Find Duplicate in Array.

Given a read-only array of n integers between 1 and n , find one number that repeats.

Input Format:

First Line - Number of elements

n Lines - n Elements

Output Format:

Element x - That is repeated

For example:

Input	Result
5 1 1 2 3 4	1

Answer: (space) regime: 0 kb

```
1 #include <stdio.h>
2
3 int main() {
4     int n;
5     scanf("%d", &n);
6
7     int A[n + 1];
8     for (int i = 0; i < n; i++) {
9         scanf("%d", &A[i]);
10    }
11
12    int freq[n + 1];
13    for (int i = 0; i < n; i++) {
14        freq[i] = 0;
15    }
16
17    for (int i = 0; i < n; i++) {
18        if (freq[A[i]] == 1) {
19            printf("%d\n", A[i]);
20            return 0;
21        }
22        freq[A[i]] = 1;
23    }
24
25    return 0;
26 }
27
```

	Input	Expected	Got	
✓	10 10 9 7 6 5 1 2 3 8 4 1	1	1	✓
✓	5 1 2 3 4 1	1	1	✓
✓	5 1 1 2 3 5	1	1	✓

Passed all tests! ✓

Correct

Marks for this submission: 1.00/1.00

Print the intersection of two sorted arrays.
 OR
 In other words,
 Given 2 sorted arrays, find all the elements which occur in both the arrays.
 Input Format:
 1. The first line contains T, the number of test cases. Following T lines contain:
 1. Line 1 contains M1, followed by M1 integers of the first array.
 2. Line 2 contains M2, followed by M2 integers of the second array.
 Output Format:
 The intersection of the arrays in a single line.

Example:

Input:
 1
 1 10 15 17
 2 7 10 15 17 200
 Output:
 10 17
 Input:
 1
 1 2 3 4 5 6
 2 1 6
 Output:
 1 6

For example:

Input	Output
1 1 10 15 17 2 2 7 10 15 17 200	10 17

Answer: (strictly within 0.7s)

```

1 // C++ implementation
2
3 #include <iostream>
4
5 using namespace std;
6
7 int* intersect(int a[], int n, int b[], int m)
8 {
9     int i = 0;
10    int j = 0;
11    while (i < n && j < m)
12    {
13        if (a[i] < b[j])
14            i++;
15        else if (a[i] > b[j])
16            j++;
17        else
18        {
19            cout << a[i] << " ";
20            i++;
21            j++;
22        }
23    }
24    return 0;
25 }
26
27 int main()
28 {
29     int a[] = {1, 10, 15, 17};
30     int n = sizeof(a) / sizeof(a[0]);
31     int b[] = {2, 7, 10, 15, 17, 200};
32     int m = sizeof(b) / sizeof(b[0]);
33     intersect(a, n, b, m);
34     return 0;
35 }

```

Input	Expected	Out
1 1 10 15 17 2 2 7 10 15 17 200	10 17	10 17
1 1 2 3 4 5 6 2 1 6	1 6	1 6

Followed by level 1

Submit

Works for the submission: 100/100

Input	Result
2	18 47
5 18 27 37	
6	
2 7 18 25 37 38	

```

1 //convolve_reshape.c
2
3 int main() {
4     int T;
5     scanf("%d", &T);
6
7     while(T--) {
8         int M;
9         scanf("%d", &M);
10        int N[M];
11        for(int i = 0; i + M - 1 < T; i++) {
12            scanf("%d", &N[i]);
13        }
14
15        int M2;
16        scanf("%d", &M2);
17        int N2[M2];
18        for(int i = 0; i + M2 - 1 < T; i++) {
19            scanf("%d", &N2[i]);
20        }
21
22        int k = M2 / 2 + 1;
23        while(k >= M2) k = M2 / 2;
24        if(N[k-1] < N2[k-1]) {
25            k++;
26        } else if(N[k-1] == N2[k-1]) {
27            k--;
28        }
29        printf("%d\n", k);
30    }
31}

```

	Input	Expected	Obs	
✓	1 5 10 15 20 25 0 0 5 10 15 20 25	10 15	10 15	✓
✓	1 5 10 15 20 25 0 0 5	1 5	1 5	✓

© 2006 The Authors
Journal compilation © 2006 Blackwell Publishing Ltd

Given an array A of sorted integers and another non-negative integer k , find if there exists 2 indices i and j such that $A[j] - A[i] = k$, $i \neq j$.

Input Format:

First Line n - Number of elements in an array

Next n lines - N elements in the array

k - Non-negative integer

Output Format:

1 - if pair exists

0 - if no pair exists

Explanation for the given Sample Testcase:

NTS as $5 - 1 = 4$

So return 1.

For example:

Input	Result
4	1
1 3 5	
4	

Answer: Complexity: $O(N)$

```

1 //File Name: sortedArray.ko
2
3 int main() {
4     int N, k;
5     scanf("%d", &N);
6
7     int A[N];
8     for (int i = 0; i < N; i++) {
9         scanf("%d", &A[i]);
10    }
11
12    scanf("%d", &k);
13
14    int i = 0, j = 1;
15    while (i < N-1 & j < N) {
16        int diff = A[j] - A[i];
17        if (diff == k & i != j) {
18            printf("%d\n", 1);
19            return 0;
20        } else if (diff < k) {
21            j++;
22        } else {
23            i++;
24            if (i == j) {
25                j++;
26            }
27        }
28    }
29    printf("%d\n", 0);
30    return 0;
31 }

```

Input	Expected	Got
4 1 3 5 4	1	1
10 1 4 6 8 12 15 19 20 23 25 3	1	1
10 1 2 3 5 11 15 16 17 20 20 0	0	0
10 0 2 3 7 11 15 16 20 21 25 10	1	1

Passed all tests! ✓

Correct

Marks for this submission: 1.00/1.00

Given an array A of sorted integers and another non-negative integer k, find if there exists 2 indices i and j such that $A[j] - A[i] = k$, $i > j$.

Input Format:

First Line n - Number of elements in an array

Next n Lines - N elements in the array

k - Non - Negative Integer

Output Format:

1 - If pair exists

0 - If no pair exists

Explanation for the given Sample Testcase:

YES as $5 - 1 = 4$

So Return 1.

For example:

Input	Result
5	1
1 4 5	
4	

Answer: (penalty regime: 0 %)

```

1 //File name: catd1a.c
2
3 int main() {
4     int n, k;
5     scanf("%d", &n);
6
7     int A[n];
8     for (int i = 0; i < n; i++) {
9         scanf("%d", &A[i]);
10    }
11
12    scanf("%d", &k);
13
14    int i = 0, j = 1;
15    while (i < n && j < n) {
16        int diff = A[j] - A[i];
17        if (diff == k && i < j) {
18            printf("Yes\n");
19            return 0;
20        } else if (diff < k) {
21            j++;
22        } else {
23            i++;
24        }
25    }
26
27    printf("No\n");
28    return 0;
29 }
30

```

Input	Expected	Got	
5 1 4 5 4	1	1	✓
10 1 4 6 8 12 17 18 20 23 25 5	1	1	✓
10 1 2 3 5 12 17 18 21 28 29 0	0	0	✓
10 0 2 3 7 12 17 18 20 23 25 10	1	1	✓

(Passed all tests) ✓

Correct

Wrote for this submission: 1.20/1.00