**71.commonElementsSum**

Read the question carefully and follow the input and output format.  
  
Given 2 integer arrays , write a program to find the sum of common elements in both the arrays.  
  
If there are no common elements print 0.  
  
**Input and Output Format :**  
First line of input consists of n, the number of elements. Next n lines correspond to the first array elements and the next n lines correspond to the second array elements. Output consist of an integer, which is the sum  
  
1) Print "Invalid array size" when size of the array is a negative number and terminate the program.  
2) Print "Invalid input" when there is any negative numbers available in the input array and terminate the program.  
  
Include a function named commonElementsSum(int elements1[],int elements2[],int size) whose return type is an integer, the sum.  
  
**Sample Input 1:**  
4  
1  
2  
3  
4  
2  
3  
6  
7  
  
**Sample Output 1:**  
5  
  
**Sample Input 2:**  
3  
8  
6  
-7  
  
**Sample Output 2:**  
Invalid input

#include<stdio.h>

#include<stdlib.h>

int common[20];

int commonElementsSum(int elements1[],int elements2[],int size);

int main(){

int size=0,input1[20],input2[20],i=0,sum=0;

scanf("%d",&size);

if(size<0){

printf("Invalid array size");

getchar();

getchar();

exit(0);

}

for(i=0;i<size;i++){

scanf("%d",&input1[i]);

if(input1[i]<0){

printf("Invalid Input");

getchar();

getchar();

exit(0);

}

}

for(i=0;i<size;i++){

scanf("%d",&input2[i]);

if(input2[i]<0){

printf("Invalid Input");

getchar();

getchar();

exit(0);

}

}

sum=commonElementsSum(input1,input2,size);

printf("%d",sum);

getchar();

getchar();

return 0;

}

int commonElementsSum(int elements1[],int elements2[],int size){

int i=0,j=0,k=0,temp=0,sum=0;

for(i=0;i<size;i++){

for(j=0;j<size;j++){

if(elements1[i]==elements2[j]){

common[k]=elements1[i];

k++;

}

}

}

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

sum=sum+common[i];

return sum;

}

**72.Palindromic Number**

Write a program to find whether the given input number is a palindrome.

Include a function named **checkPalindrome** that accepts an integer and returns an integer. The function returns

1. 1 if the input is a palindrome
2. 0 if the input is not a palindrome
3. -1 if the input is a negative number

Print Invalid Input if the function returns -1.

**Input and Output Format:**

Input consists of a single integer.

Refer sample output for formatting specifications.

**Sample Input 1:**

2002

**Sample Output 1:**

yes

**Sample Input 2:**

167

**Sample Output 2:**

no

**Sample Input 3:**

**-**2345

**Sample Output 3:**

Invalid Input

#include<stdio.h>

#include<stdlib.h>

int checkPalindrome (int);

int main(){

int number=0,result;

scanf("%d",&number);

result = checkPalindrome (number);

if(result==1)

printf("yes");

else if(result==0)

printf("no");

else

printf("Invalid input");

getchar();

getchar();

return 0;

}

int checkPalindrome (int n){

int res=0,temp=0,rem=0,sum=0,reverse=0;

if(n<0)

res=-1;

else{

temp=n;

while(temp!=0){

rem=temp%10;

reverse=reverse\*10+rem;

temp/=10;

}

if(reverse==n)

res=1;

else

res=0;

}

return res;

}

**73.sumEvenIndex**

Read the question carefully and follow the input and output format.  
  
Write a program to find the sum of the indexes (positions) of even numbers in the Array. Consider 0 index as 1 and 1 index is 2 and so on……  
Note : Assume Array Index Starts From 1  
  
**Input and Output Format :**  
First line of input consists of n, the number of elements. Next n lines correspond to the array elements. Output consist of an integer, which is the sum.  
  
1) Print "Invalid array size" when size of the array is a negative number and terminate the program.  
2) Print "Invalid input" when there is any negative number available in the input array and terminate the program.  
  
Include a function named sumEvenIndex(int numbers[], int size) whose return type is an integer, which is the sum..  
**Sample Input 1:**  
7  
4  
2  
7  
9  
1  
10  
13  
  
**Sample Output 1:**  
9  
  
**Sample Input 2:**  
-13  
  
**Sample Output 2:**  
Invalid array size

#include<stdio.h>

#include<stdlib.h>

int sumEvenIndex(int numbers[], int size);

int main(){

int n=0,sum=0,input[30],i,flag=0;

scanf("%d",&n);

if(n<0){

printf("Invalid array size");

getchar();

getchar();

exit(0);

}

for(i=1;i<=n;i++){

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

if(input[i]<0){

printf("Invalid Input");

getchar();

getchar();

exit(0);

}

}

sum = sumEvenIndex(input,n);

printf("%d",sum);

getchar();

getchar();

return 0;

}

int sumEvenIndex(int numbers[], int size){

int i=0,sum=0;

for(i=1;i<=size;i++){

if(numbers[i]%2==0)

sum=sum+i;

}

return sum;

}

**74.sumEvenOddProduct**

Read the question carefully and follow the input and output format.  
  
Write a program to find the sum of  product of even digits  and product of odd digits in a given number.  
  
If number contains only even numbers or odd numbers take the other numbers product as 1.  
  
**Input and Output Format :**  
Input consists of a single integer. Output consist of the sum of even digit product and odd digit product.  
  
Print "Number too large" when the given input number is greater than 32767  
Print "Number too small" when the given input number is a negative number.  
  
Include a function named sumEvenOddProduct(int number) whose return type is integer, the sum  
  
**Sample Input 1:**  
4564  
**Sample Output 1:**  
101  
  
{Hint :   (4\*6\*4) + (5)  =  96 +5 = 101}  
**Sample Input 2:**  
1357  
**Sample Output 2:**  
106  
  
  
**Sample Input 3:**  
981357  
**Sample Output 3:**  
Number too large

#include<stdio.h>

#include<stdlib.h>

int sumEvenOddProduct(int number);

int main(){

int number,result=0;

scanf("%d",&number);

if(number>32767)

printf("number too large");

else if(number<0)

printf("number too small");

else

{

result = sumEvenOddProduct(number);

printf("%d",result);

}

getchar();

getchar();

return 0;

}

int sumEvenOddProduct(int number){

int res=0,i,rem=0,evenprod=1,oddprod=1;

while(number!=0){

rem=number%10;

if(rem%2==0)

evenprod=evenprod\*rem;

else

oddprod=oddprod\*rem;

number=number/10;

}

res = evenprod + oddprod;

return res;

}

**75.Sum of the Digits**

In a lucky draw everybody got one coupon with some code. They need to sum the digits in the code and send SMS to the given number. Write a program to find the sum of digits in a number.

Include a function named **sumDigits**that accepts an integer argument and returns an integer that corresponds to the sum of the digits. The function returns -1 if the input is less than zero or if the roll number is greater than 32767.

If the function returns -1, print “Invalid Input”.

**Input and Output Format:**

The input consists of an integer.

The output consists of an integer that corresponds to the sum of the digits in the number.

**Sample Input 1:**

3487

**Sample Ouput 1:**

22

**Sample Input 2:**

-8

**Sample Output 2:**

Invalid Input

#include<stdio.h>

#include<stdlib.h>

int sumDigits(int);

int main(){

int number,result=0;

scanf("%d",&number);

result = sumDigits(number);

if(result==-1)

printf("Invalid input");

else

printf("%d",result);

getchar();

getchar();

return 0;

}

int sumDigits(int n){

int res=0,i,rem=0;

if(n<0 || n>32767)

res = -1;

else{

while(n!=0){

rem=n%10;

res=res+rem;

n=n/10;

}

}

return res;

}

**76.Sum of squares of prime numbers**

Given an integer n, write a program to find the sum of squares of prime numbers upto and including n.

Include a function named **sumSquarePrime**that accepts an integer argument and returns an integer that corresponds to result. The function returns -1 if the input is less than zero or if the number is greater than 32767.

If the function returns -1, print “Invalid Input”.

Please note that 1 is neither prime nor composite.

**Input and Output Format:**

The input consists of an integer.

The output consists of an integer that corresponds to the sum of the squares of prime numbers.

**Sample Input 1:**

10

**Sample Ouput 1:**

87

**Sample Input 2:**

-8

**Sample Output 2:**

Invalid Input

#include<stdio.h>

#include<stdlib.h>

int sumSquarePrime(int);

int main(){

int number,result=0;

scanf("%d",&number);

result = sumSquarePrime(number);

if(result==-1)

printf("Invalid input");

else

printf("%d",result);

getchar();

getchar();

return 0;

}

int sumSquarePrime(int n){

int i,count=0,num,sum=0;

for(num = 1;num<=n;num++){

count=0;

for(i=1;i<=num;i++){

if(num%i==0)

count++;

}

if(count==2)

sum=sum+(num\*num);

}

return sum;

}

**77.3/5 Number**

Write a program to find whether the given number is a 3/5 Number.

A number is a 3/5 Number if the product of the digits in the number is divisible by 3 or 5.

Include a function named **divisibleByThreeFive** that accepts an integer argument and returns an integer. The function returns

1. 1 if it is a 3/5 Number
2. 0 if it is not a 3/5 Number
3. -1 if it is a negative number

**Input and Output Format:**

Input consists of a single integer.

Output consists of a string.

Refer sample output for formatting specifications.

**Sample Input 1:**

251

**Sample Output 1:**

yes

**Sample Input 2:**

241

**Sample Output 2:**

no

**Sample Input 3:**

-9

**Sample Output 3:**

Invalid Input

#include<stdio.h>

#include<stdlib.h>

int divisibleByThreeFive(int);

int main(){

int number=0,result;

scanf("%d",&number);

result = divisibleByThreeFive(number);

if(result==1)

printf("yes");

else if(result==0)

printf("no");

else

printf("Invalid input");

getchar();

getchar();

return 0;

}

int divisibleByThreeFive(int n){

int res=1,i,rem=0,prod=1;

if(n<0)

res = -1;

else{

while(n!=0){

rem=n%10;

prod=prod\*rem;

n=n/10;

}

if(prod%3==0 || prod%5==0)

res=1;

else

res=0;

}

return res;

}

**78.aboveAverageMarks**

Read the question carefully and follow the input and output format.  
  
Given an input array that represents the marks of students, find out the marks which are greater than or equal to average mark of all students.  
  
**Input and Output Format:**  
First line of input consists of n, the number of elements in the input array.  
Next n lines correspond to the array elements. Output consist of an integer array.  
  
1) Print "Invalid array size" when size of the array is negative and terminate the program.  
2) Print "Invalid input" when there is any negative numbers available in the input array and terminate the program.  
  
Include a function named aboveAverageMarks(int array[], int size) whose return type is void.  
The output array is stored in a global variable named above\_average.  
  
**Sample Input 1:**  
5  
10  
20  
30  
40  
50  
**Sample Output 1:**  
30  
40  
50  
  
**Sample Input 2:**  
4  
-3  
**Sample Output 2:**  
Invalid Input

#include<stdio.h>

#include<stdlib.h>

void aboveAverageMarks(int array[], int size);

int main(){

int size=0,input1[20],i=0;

scanf("%d",&size);

if(size<0){

printf("Invalid input");

getchar();

getchar();

exit(0);

}

for(i=0;i<size;i++){

scanf("%d",&input1[i]);

if(input1[i]<0){

printf("Invalid Input");

getchar();

getchar();

exit(0);

}

}

aboveAverageMarks(input1,size);

return 0;

}

void aboveAverageMarks(int array[], int size){

int i=0,j=0,avg=0,sum=0;

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

sum=sum+array[i];

avg=sum/size;

for(i=0;i<size;i++){

if(array[i]>=avg)

printf("%d\n",array[i]);

}

getchar();

getchar();

}

**79.changeNumber**

Read the question carefully and follow the input and output format.  
  
Tom needs to generate a new number from the given input with the following conditions.Consider Input is always a 3 digit number.  
  
conditions:  
(i) Middle digit comes first.  
(ii) Last digit should come in middle  
(iii) First digit should come as a last digit

Business rule:  
1. Print "Invalid input" if input is negative number.  
2. Print "Not a 3 digit number" if the given number is not a 3 digit number.  
158

581  
Include a function named changeNumber(int number) that returns an integer.  
  
**Input and Output Format:**  
Input consists of an integer.  
Refer business rules and sample output for output format.  
  
**Sample Input 1:**  
123  
**Sample Output 1:**  
231  
  
  
**Sample Input 2:**  
1234  
**Sample Output 2:**  
Not a 3 digit number

#include<stdio.h>

#include<stdlib.h>

int changeNumber(int n);

int main(){

int number=0,result;

scanf("%d",&number);

if(number<0)

printf("invalid input");

else if(number<100 || number>999)

printf("Not a three digit number");

else{

result = changeNumber(number);

printf("%d",result);

}

getchar();

getchar();

return 0;

}

int changeNumber(int n){

int rem=0,d1,d2,d3,i=1,temp=1,res=0;

do{

rem=n%10;

d3=rem;

n=n/10;

rem=n%10;

d2=rem;

n=n/10;

rem=n%10;

d1=rem;

}while(0);

do{

res=res+(d1\*i);

i=i\*10;

res=res+(d3\*i);

i=i\*10;

res=res+(d2\*i);

}while(0);

return res;

}

**80.differentElements**

Read the question carefully and follow the input and output format.  
  
Given two input arrays find out the elements which are not common.  
  
**Input and Output Format:**  
  
First line of input consists of n, the number of elements. Next n lines correspond to the first array elements and the next n lines correspond to the second array elements. Output consist of an integer array, which contains the elements that are not common between the first and second array.  
  
1) Print Invalid array size when size of the array is a negative number and terminate the program.  
2) Print Invalid input when there is any negative numbers available in the input array and terminate the program.  
  
Include a function named differentElements(int set1[], int set2[], int size) whose return type is void.  
The output array is stored in a global variable named not\_common.  
  
**Sample Input 1:**  
5  
1 2 3 4 5  
5 6 4 8 7  
  
**Sample Output 1:**  
1  
2  
3  
6  
8  
7  
  
**Sample Input 2:**  
5  
1 4 8 9 4  
-8  
**Sample Output 2:**  
Invalid input

#include<stdio.h>

#include<stdlib.h>

int not\_common[20];

void differentElements(int set1[], int set2[], int size);

int main(){

int size=0,input1[20],input2[20],i=0;

scanf("%d",&size);

if(size<0){

printf("Invalid Array size");

getchar();

getchar();

exit(0);

}

for(i=0;i<size;i++){

scanf("%d",&input1[i]);

if(input1[i]<0){

printf("Invalid Input");

getchar();

getchar();

exit(0);

}

}

for(i=0;i<size;i++){

scanf("%d",&input2[i]);

if(input2[i]<0){

printf("Invalid Input");

getchar();

getchar();

exit(0);

}

}

differentElements(input1,input2,size);

return 0;

}

void differentElements(int set1[], int set2[], int size){

int i=0,j=0,k=0,flag=0,flag1=0;

for(i=0;i<size;i++){

flag=0;

for(j=0;j<size;j++){

if(set1[i]==set2[j])

flag=1;

}

if(flag==0){

not\_common[k]=set1[i];

k++;

}

}

for(i=0;i<size;i++){

flag1=0;

for(j=0;j<size;j++){

if(set2[i]==set1[j])

flag1=1;

}

if(flag1==0){

not\_common[k]=set2[i];

k++;

}

}

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

printf("%d\n",not\_common[i]);

getchar();

getchar();

}