**31.sumPrimeArray**

Read the question carefully and follow the input and output format.  
  
John is working in a bank. He has created account details transaction in a file and protected it with a password. He sent the file to his manager for review. The file is protected with a password. The password is the sum of Prime numbers. Write a function to generate the password.  
  
**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.  
3) Print 0, when there are no prime numbers in a given input array.  
  
Include a function named sumPrimeArray(int array[], int size) whose return type is an integer, which is the prime sum.  
  
**Sample Input 1:**  
5  
1  
2  
3  
4  
5  
  
**Sample Output 1:**  
10  
  
**Sample Input 2:**  
3  
4  
8  
9  
  
**Sample Output 2:**  
0

#include<stdio.h>

#include<stdlib.h>

int sumPrimeArray(int array[], int size);

int main(){

int n=0,flag=0,i,input[20],sum=0;

scanf("%d",&n);

if(n<0){

printf("Invalid Input");

getchar();

getchar();

exit(0);

}

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

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

if(input[i]<0)

flag=1;

}

if(flag==1){

printf("Invalid Input");

getchar();

getchar();

exit(0);

}

sum = sumPrimeArray(input,n);

printf("%d",sum);

getchar();

getchar();

return 0;

}

int sumPrimeArray(int array[],int n){

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

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

count=0;

for(j=1;j<=array[i];j++){

if(array[i]%j==0)

count++;

}

if(count==2)

sum=sum+(array[i]);

}

return sum;

}

**32. avgOddKeyValues**

Read the question carefully and follow the input and output format.  
  
  
Given an input array, First index represents key and second index represents the value and so on... Write code to find out the average of all values whose keys are odd numbers.  
  
**Input and Output Format :**  
First line of input consists of n, the next n lines correspond to the elements of the array. Output consist of the an integer.  
  
Print "Invalid array size" when size of the array is a negative number and terminate the program.  
Print "Invalid input" when there is any negative number available in the input array and terminate the program.  
  
Include a function named avgOddKeyValues(int numbers[], int size) whose return type is an integer.  
  
**Sample Input 1:**  
8  
1  
3  
2  
4  
3  
16  
4  
25  
**Sample Output 1:**  
9  
  
**Sample Input 2:**  
-3  
**Sample Output 2:**  
Invalid array size

#include<stdio.h>

#include<stdlib.h>

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

int main(){

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

scanf("%d",&n);

if(n<0){

printf("Invalid array size");

getchar();

getchar();

exit(0);

}

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

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

if(input[i]<0)

flag=1;

}

if(flag==1){

printf("Invalid Input");

getchar();

getchar();

exit(0);

}

avg = avgOddKeyValues(input,n);

printf("%d",avg);

getchar();

getchar();

return 0;

}

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

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

for(i=0;i<size;i=i+2){

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

sum=sum+numbers[i+1];

j++;

}

}

avg = sum/j;

return avg;

}

**33.Product of MaxMin Element**

Write a program to find the product of the maximum and minimum element in a given input array.

Include a function named **productOfMaxMin** that accepts 2 arguments and returns an int. The first argument is the input array and the second argument is an int that corresponds to the size of the array. The function returns an int that corresponds to the product of maximum and minimum element.

If the size of the array is negative or if any element in the array is negative, print “Invalid Input” and terminate the program.

**Input and Output Format:**

Input consists of n+1 integers. The first integer corresponds to n, the number of elements in the array. The next 'n' integers correspond to the elements in the array.

Output consists of an integer that corresponds to the product of maximum and minimum element in the array.

Assume that the maximum number of elements in the array is 20.

**Sample Input 1:**

8

2

12

3

4

6

8

10

9

**Sample Output 1:**

24

**Sample Input 2:**

-5

**Sample Output 2:**

Invalid Input

**Sample Input 3:**

5

23

2

-200

**Sample Output 3:**

Invalid Input

#include<stdio.h>

#include<stdlib.h>

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

int main(){

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

scanf("%d",&n);

if(n<0){

printf("Invalid input");

getchar();

getchar();

exit(0);

}

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

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

if(input[i]<0)

flag=1;

}

if(flag==1){

printf("Invalid Input");

getchar();

getchar();

exit(0);

}

product = productOfMaxMin(input,n);

printf("%d",product);

getchar();

getchar();

return 0;

}

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

int i,j=0,prod=0,temp=0;

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

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

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

temp = numbers[j];

numbers[j] = numbers[i];

numbers[i] = temp;

}

}

}

prod = numbers[0] \* numbers[size-1];

return prod;

}

**34.registerAccountNumbers**

Read the question carefully and follow the input and output format.  
  
Given an array in which the elements are in xxxyy format, where first xxx digits represent the Branch code and the yy represents the account  
id.  Find out the No of accounts in the given branch code  
  
Input and Output Format :  
The first input n corresponds to the size of the array, the next n lines correspond to the elements of the array and the last line of the input corresponds to the branch code.  
Output corresponds to the number of accounts in the given branch code  
If the given branch code is not available, print 0.  
  
1) Print "Invalid array size" when size of the array is a negative number and terminate the program  
2) Print "Invalid account Number" when there is any negative number available in the input array and terminate the program  
3) Print "Invalid branch code" when branch code is negative number and terminate the program  
  
Include a function named registerAccountNumbers (int size, int account\_numbers[], int branch\_code) that returns the no of accounts  
  
**Sample Input 1 :**  
6  
12345  
12370  
12324  
13355  
13333  
14575  
123  
  
**Sample Output 1 :**  
3  
  
**Sample Input 2 :**  
-6  
  
**Sample Output 2:**  
Invalid array size

#include<stdio.h>

#include<stdlib.h>

int registerAccountNumbers (int size, int account\_numbers[], int branch\_code);

int main(){

int n=0,no\_of\_acc=0,input[30],i,flag=0,bcode=0;

scanf("%d",&n);

if(n<0){

printf("Invalid array size");

getchar();

getchar();

exit(0);

}

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

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

if(input[i]<0)

flag=1;

}

if(flag==1){

printf("Invalid account number");

getchar();

getchar();

exit(0);

}

scanf("%d",&bcode);

if(bcode<0){

printf("Invalid branch code");

getchar();

getchar();

exit(0);

}

no\_of\_acc=registerAccountNumbers(n,input,bcode);

printf("%d",no\_of\_acc);

getchar();

getchar();

return 0;

}

int registerAccountNumbers(int size, int account\_numbers[], int branch\_code){

int count=0,i,rem,temp=0,k=1,bcode=0,x=1;

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

temp=account\_numbers[i];

bcode=temp/100;

if(bcode==branch\_code)

count++;

}

return count;

}

**35.newArraySum**

Read the question carefully and follow the input and output format.  
  
Given an input array which contains age of some employees, write a program t fund the sum of ages of employees greater than 18.  
  
**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 newArraySum(int age[],int size) whose return type is an integer, which is the sum.  
  
**Sample Input 1:**  
5  
21  
22  
17  
10  
25  
**Sample Output 1:**  
68  
  
**Sample Input 2:**  
6  
50  
-36  
  
**Sample Output 2:**  
Invalid input

#include<stdio.h>

#include<stdlib.h>

int newArraySum(int age[],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=0;i<n;i++){

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

if(input[i]<0)

flag=1;

}

if(flag==1){

printf("Invalid Input");

getchar();

getchar();

exit(0);

}

sum = newArraySum(input,n);

printf("%d",sum);

getchar();

getchar();

return 0;

}

int newArraySum(int age[],int size) {

int i=0,sum=0;

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

if(age[i]>18)

sum=sum+age[i];

}

return sum;

}

**36.Array Product**

Write a program to find the product of posive/nonnegative elements in a given array.

Include a function named **calculateProduct** that accepts 2 arguments and returns an int. The first argument is the input array and the second argument is an int that corresponds to the size of the array. The function returns an int that corresponds to the product.

If the size of the array is negative or if it is greater than 10 or if any element in the array is more than 2 digits, print “Invalid Input” and terminate the program.

**Input and Output Format:**

Input consists of n+1 integers. The first integer corresponds to n, the number of elements in the array. The next 'n' integers correspond to the elements in the array.

Output consists of an integer that corresponds to the product of positive numbers in the array.

**Sample Input 1:**

8

1

-2

3

4

-6

8

10

-6

**Sample Output 1:**

960

**Sample Input 2:**

-5

**Sample Output 2:**

Invalid Input

**Sample Input 3:**

5

23

2

200

**Sample Output 3:**

Invalid Input

#include<stdio.h>

#include<stdlib.h>

int calculateProduct(int[],int);

int main(){

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

scanf("%d",&n);

if(n<0 || n>10 ){

printf("Invalid Input");

getchar();

getchar();

exit(0);

}

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

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

if(input[i]>=100 || input[i]<=-100)

flag=1;

}

if(flag==1){

printf("Invalid Input");

getchar();

getchar();

exit(0);

}

prod = calculateProduct(input,n);

printf("%d",prod);

getchar();

getchar();

return 0;

}

int calculateProduct(int number[],int size) {

int i=0,prod=1;

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

if(number[i]>0)

prod=prod\*number[i];

}

return prod;

}

**37.Perfect Number**

Write a program to find whether the given number is a perfect Number.

A number is a perfect number if the sum of the proper divisors of the number is equal to the number itself.

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

1. 1 if the input is a Perfect Number
2. 0 if the input is not a Perfect Number
3. -1 if the input is a negative number or if it is greater than 32767

**Input and Output Format:**

Input consists of a single integer.

Output consists of a string.

Refer sample output for formatting specifications.

**Sample Input 1:**

6

**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 findPerfect(int);

int main(){

int number,result=0;

scanf("%d",&number);

result = findPerfect(number);

if(result==1)

printf("yes");

else if(result==0)

printf("no");

else

printf("Invalid input");

getchar();

getchar();

return 0;

}

int findPerfect(int n){

int res=0,i,sum=0;

if(n<0 || n>32767)

res = -1;

else{

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

if(n%i==0)

sum=sum+i;

}

if(sum==n)

res = 1;

else

res=0;

}

return res;

}

**38.Squares of Even Digits**

Write a program to find the sum of the squares of even digits in a number.

Include a function named sumSquareEven that accepts an integer argument and returns an integer . The function returns -1 if the number is less than zero or if the number is greater than 32767.

Print Invalid Input if the function returns -1.

**Input and Output Format:**

Input consists of an integer.

Output consists of an integer.

Refer sample output for formatting specifications.

**Sample Input 1:**

3487

**Sample Ouput 1:**

80

**Sample Input 2:**

-8

**Sample Output 2:**

Invalid Input

#include<stdio.h>

#include<stdlib.h>

int sumSquareEven(int);

int main(){

int number,result=0;

scanf("%d",&number);

result = sumSquareEven(number);

if(result==-1)

printf("Invalid input");

else

printf("%d",result);

getchar();

getchar();

return 0;

}

int sumSquareEven(int n){

int res=0,i,rem=0;

if(n<0 || n>32767)

res = -1;

else{

while(n!=0){

rem=n%10;

if(rem%2==0)

res=res+(rem\*rem);

n=n/10;

}

}

return res;

}

**39.sumPrimeArray**

Read the question carefully and follow the input and output format.  
  
John is working in a bank. He has created account details transaction in a file and protected it with a password. He sent the file to his manager for review. The file is protected with a password. The password is the sum of Prime numbers. Write a function to generate the password.  
  
**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.  
3) Print 0, when there are no prime numbers in a given input array.  
  
Include a function named sumPrimeArray(int array[], int size) whose return type is an integer, which is the prime sum.  
  
**Sample Input 1:**  
5  
1  
2  
3  
4  
5  
  
**Sample Output 1:**  
10  
  
**Sample Input 2:**  
3  
4  
8  
9  
  
**Sample Output 2:**  
0

#include<stdio.h>

#include<stdlib.h>

int sumPrimeArray(int array[], int size);

int main(){

int n=0,flag=0,i,input[20],sum=0;

scanf("%d",&n);

if(n<0){

printf("Invalid Input");

getchar();

getchar();

exit(0);

}

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

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

if(input[i]<0)

flag=1;

}

if(flag==1){

printf("Invalid Input");

getchar();

getchar();

exit(0);

}

sum = sumPrimeArray(input,n);

printf("%d",sum);

getchar();

getchar();

return 0;

}

int sumPrimeArray(int array[],int n){

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

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

count=0;

for(j=1;j<=array[i];j++){

if(array[i]%j==0)

count++;

}

if(count==2)

sum=sum+(array[i]);

}

return sum;

}

**40. Array Multiplication in Reverse**

A company wanted to know the reward points of the employee so that at the end of every month they will credit some amount along with their salary. Each employee has 2 separate lists, in first list records will be sorted in employee’s employee number in ascending order. Second list records will be sorted in employee’s employee number in descending order. Hence the management has decided to multiply both the reward points and credit the amount based on the points. Here they followed the formula for multiplying the first entry value in the first list with the last entry value in the second list and second entry from the first list with the second last record from the second list. Repeat the same for all the entries in the lists.

Include a function named **arrayProduct** that accepts 3 arguments and ints return type is void. The first argument is the input array 1, the second argument is  the input array 2 and the third argument is an int that corresponds to the size of the array. The output array is stored in a global variable named output1.

If the size of the array is negative or if any element in any of the array is negative , print “Invalid Input” and terminate the program.

**Input and Output Format:**

Input consists of utmost 2n+1 integers. The first integer corresponds to n, the number of elements in the array. The next 'n' integers correspond to the elements in the first array. The last 'n' integers correspond to the elements in the second array. If any of the inputs are invalid, then terminate the program.

Output consists of n integers that correspond to the elements in the result array.

Assume that the maximum size of the array is 20.

**Sample Input 1 :**

5

23

2

5

32

76

2

2

21

42

4

**Sample Output 1 :**

92

84

105

64

152

**Sample Input 2:**

-5

**Sample Output 2:**

Invalid Input

**Sample Input 3:**

5

23

2

-5

**Sample Output 3:**

Invalid Input

#include<stdio.h>

#include<stdlib.h>

int output1[20];

void arrayProduct(int[],int[],int);

int main(){

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

int flag1=0,flag2=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)

flag1=1;

}

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

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

if(input2[i]<0)

flag2=1;

}

if(flag1==1 || flag2==1){

printf("Invalid Input");

getchar();

getchar();

exit(0);

}

arrayProduct(input1,input2,size);

return 0;

}

void arrayProduct(int ar1[],int ar2[],int size){

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

for(i=0,j=size-1;i<size&&j>=0;i++,j--){

output1[i] = ar1[i]\*ar2[j];

printf("%d\n",output1[i]);

}

getchar();

getchar();

}