**51.sumTwoFive**

Read the question carefully and follow the input and output format.  
  
Given an integer array find the sum of elements which end with 2 or 5.  
  
**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, the sum.  
  
1) Print "Invalid array size" when size of the array is negative .  
2) Print "Invalid input" when there is any negative number available in the input array and terminate the program.  
  
Include a function named sumTwoFive(int array[], int size) whose return type is an integer, the sum  
  
**Sample Input 1:**  
5  
22  
35  
5  
2  
10  
  
**Sample Output 1:**  
64  
  
**Sample Input 2:**  
-6  
  
**Sample Output 2:**  
Invalid array size

#include<stdio.h>

#include<stdlib.h>

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

int main(){

int n=0,input[20],i,sum=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){

printf("Invalid Input");

getchar();

getchar();

exit(0);

}

}

sum = sumTwoFive(input,n);

printf("%d",sum);

getchar();

getchar();

return 0;

}

int sumTwoFive(int array[], int size){

int i,rem,temp,sum=0;

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

temp = array[i];

rem = temp % 10;

if(rem==2 || rem==5)

sum = sum+array[i];

}

return sum;

}

**52.Sum of Odd Digits**

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

Include a function named **sumOddDigits**that accepts an integer argument and returns an integer that corresponds to the sum of the odd digits. The function returns -1 if the input is less than zero or if it 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 odd digits in the number.

**Sample Input 1:**

3487

**Sample Ouput 1:**

10

**Sample Input 2:**

-8

**Sample Output 2:**

Invalid Input

#include<stdio.h>

#include<stdlib.h>

int sumOddDigits(int);

int main(){

int num=0,res=0;

scanf("%d",&num);

res = sumOddDigits(num);

if(res==-1)

printf("Invalid input");

else

printf("%d",res);

getchar();

getchar();

return 0;

}

int sumOddDigits(int number){

int sum=0,rem=0;

if(number<0 || number>32767)

sum=-1;

else{

while(number!=0){

rem=number%10;

if(rem%2!=0)

sum=sum+rem;

number=number/10;

}

}

return sum;

}

**53.LCM**

Write a program to calculate the LCM of the 2 given integers.

Include a function named **calculateLCM** that accepts 2 integer arguments and returns an int that corresponds to the LCM of the 2 numbers.

Print Invalid Input and terminate the program in the following cases:

1. Any of the 2 inputs is greater than 1000
2. Any of the 2 inputs is negative

**Input and Output Format:**

Input consists of 2 integers.

Output consists of a single integer that corresponds to the LCM.

**Sample Input 1:**

10

8

**Sample Output 1:**

40

**Sample Input 2:**

50000

**Sample Output 2:**

Invalid Input

#include<stdio.h>

#include<stdlib.h>

#include<string>

int calculateLCM(int,int);

int main(){

int a,b,res=0;

scanf("%d%d",&a,&b);

if(a>1000 || b>1000 || a<0 || b<0)

{

printf("Invalid input");

exit(0);

getchar();

getchar();

}

res=calculateLCM(a,b);

printf("%d",res);

getchar();

getchar();

return 0;

}

int calculateLCM(int num1,int num2){

int max=0;

max=(num1>num2) ? num1 : num2;

while(1){

if(max%num1==0 && max%num2==0){

break;

}

++max;

}

return max;

}

**54.Product of Prime Digits**

Write a program to find the product of the prime digits in the given input number.

Include a function named **productPrimeDigits** that accepts an integer argument and returns an integer that corresponds to the product of the prime digits in the integer.

The function returns -1 if the input number is negative or greater than 32767.

If the function returns -1, print Invalid Input.

Please note that 1 and 0 are neiher prime nor composite.

**Input and Output Format:**

Input consists of an integer.

Output consists of an integer.

Refer sample output for formatting specifications.

**Sample Input 1:**

324

**Sample Output 1:**

6

**Sample Input 2:**

-67

**Sample Output 2:**

Invalid Input

#include<stdio.h>

#include<stdlib.h>

#include<string>

int productPrimeDigits(int);

int main(){

int digit=0,product=0;

scanf("%d",&digit);

product=productPrimeDigits(digit);

if(product==-1)

printf("Invalid input");

else

printf("%d",product);

getchar();

getchar();

return 0;

}

int productPrimeDigits(int number){

int prod=1,rem=0,i,count=0,flag=0;

if(number<0 || number>32767)

prod = -1;

else

{

while(number!=0){

rem=number%10;

count=0;

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

if(rem%i==0)

count++;

}

if(count==2){

flag=1;

prod = prod\*rem;

}

number=number/10;

}

if(flag==0)

prod = 0;

}

return prod;

}

**55.dailyAllowance**

Read the question carefully and follow the input and output format.  
  
A Sales person daily allowances calculated as follows .  
Item              Money (rupees)  
Shirt                   15  
Saree                 10  
other items          5  
  
Given an input array in which the first index represents no.of shirts sold, second index represents the no of sarees sold  and the third index represents the other items sold for a particular day, Calculate the total allowances.  
  
Inlcude a function named dailyAllowance(int items[], int size) that returns an integer, which is the total allowances.  
  
Business Rules:  
1) Print "Invalid array size" when size of the array is a negative number and terminate the program  
2) Print "Invalid item count" when there is any negative numbers available in the input array and terminate the program  
3) Print "Array size greater than 3"  when size of the array is greater than 3 and terminate the program.  
  
**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 the total allowance.  
  
**Sample Input 1**:  
3  
10  
5  
10  
**Sample Output 1:**  
250  
  
**Sample Input 2:**  
4  
  
**Sample Output 2:**  
Array size greater than 3

#include<stdio.h>

#include<stdlib.h>

int dailyAllowance(int items[], int size);

int main(){

int n=0,input[20],i,allowance=0;

scanf("%d",&n);

if(n<0){

printf("Invalid array size");

getchar();

getchar();

exit(0);

}

if(n>3){

printf("Array size greater than 3");

getchar();

getchar();

exit(0);

}

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

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

if(input[i]<0){

printf("Invalid item count");

getchar();

getchar();

exit(0);

}

}

allowance = dailyAllowance(input,n);

printf("%d",allowance);

getchar();

getchar();

return 0;

}

int dailyAllowance(int items[], int size){

int shirt=15,saree=10,other=5,allowance=0;

allowance = (items[0]\*shirt) + (items[1]\*saree) + (items[2]\*other);

return allowance;

}

**56.generateNumber**

Read the question carefully and follow the input and output format.  
  
Given an input number keep the prime digits as it is and the remaining digit in given input number should be replaced with next number(add +1 to the digit).  
  
Print "Number too small" if the number is less than 0  
Print "Number too large" if the number is greater than 32767  
note : If any digit in the given input is 9 replace with 0. Consider 1 is not a prime number.  
  
Include a function generateNumber(int number) that returns the generated number  
  
**Input and Output Format :**  
Input is a single integer.  
Output is the generated number  
  
**Sample Input 1:**  
6234  
  
**Sample Output 1:**  
7235  
  
**Sample Input 2:**  
32768  
  
**Sample Output 2:**  
Number too large

#include<stdio.h>

#include<stdlib.h>

int generateNumber(int number);

int main(){

int num=0,res=0;

scanf("%d",&num);

if(num<0)

printf("number too small");

else if(num>32767)

printf("number too large");

else

{

res = generateNumber(num);

printf("%d",res);

}

getchar();

getchar();

return 0;

}

int generateNumber(int number){

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

while(number!=0){

rem=number%10;

if(rem==2 || rem==3 || rem==5 || rem==7){

res=res+(rem\*i);

i=i\*10;

}

else if(rem==9){

res=res+0;

i=i\*10;

}

else{

res=res+((rem+1)\*i);

i=i\*10;

}

number = number/10;

}

return res;

}

**57.sortCommonElements**

Read the question carefully and follow the input and output format.  
  
Find out the common elements in the given input arrays and sort the common elements in ascending order.  
  
**Input and Output Format :**  
First line corresponds to n, the size of the array. The next n lines correspond to elements in the first array. The next n lines correspond to the elements in the second array. Output corresponds to the common elements sorted in ascending order.  
  
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 sortCommonElements(int set1[],int set2[],int size) whose return type is void.  
The output array is stored in a global variable named common.  
  
**Sample Input 1:**  
5  
1  
3  
2  
5  
7  
4  
5  
6  
7  
2  
  
**Sample Output 1:**  
2  
5  
7  
  
**Sample Input 2:**  
5  
5  
8  
-9  
  
**Sample Output 2:**  
Invalid input  
  
**Sample Input 3:**  
-4  
  
**Sample Output 3:**  
Invalid array size

#include<stdio.h>

#include<stdlib.h>

int common[20];

void sortCommonElements(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 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);

}

}

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

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

if(input2[i]<0){

printf("Invalid Input");

getchar();

getchar();

exit(0);

}

}

sortCommonElements(input1,input2,size);

return 0;

}

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

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

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

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

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

common[k]=set1[i];

k++;

}

}

}

/\* Sorting elements \*/

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

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

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

temp = common[i];

common[i] = common[j];

common[j] = temp;

}

}

}

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

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

getchar();

getchar();

}

**58.findEmployeeID**

Read the question carefully and follow the input and output format.  
  
In a given input array, elements are given in this Format PPPII, where PPP represents the Project Code and II represents employee id . Find out the Employee Ids who are working in the given project code.  
  
**Input and Output Format :**  
First line of input consists of n, the number of elements. Next n lines correspond to the array elements.  The next line corresponds to the project code. Output consists of the employee id's.  
  
1) Print "Invalid array size" when size of the array is a negative number and terminate the program  
2) Print "Invalid employee id" when there is any negative numbers available in the input array and terminate the program  
3) Print "Invalid project code" when branch code is negative number and terminate the program  
  
Include a function named findEmployeeID(int size,int employee\_ids[],int project\_code). Its return type is void.  
The output array is stored in a global variable named working\_employee.  
  
**Sample Input 1:**  
5  
12345  
12334  
23457  
23478  
12546  
123  
**Sample Output 1:**  
45  
34  
  
**Sample Input 2:**  
5  
12345  
12334  
23457  
-23478  
**Sample Output 2:**  
Invalid employee id

#include<stdio.h>

#include<stdlib.h>

int working\_employee[10];

void findEmployeeID(int size,int employee\_ids[],int project\_code);

int main(){

int n=0,input[30],i,pcode=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){

printf("Invalid employee id");

getchar();

getchar();

exit(0);

}

}

scanf("%d",&pcode);

if(pcode<0){

printf("Invalid project code");

getchar();

getchar();

exit(0);

}

findEmployeeID(n,input,pcode);

return 0;

}

void findEmployeeID(int size,int employee\_ids[],int project\_code){

int count=0,i,rem,temp=0,k=1,pcode=0,x=1,num=0,var=1;

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

temp=employee\_ids[i];

pcode=temp/100;

if(pcode==project\_code){

working\_employee[count]=employee\_ids[i]%100;

count++;

}

}

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

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

getchar();

getchar();

}

**59.Array Addition in Reverse**

In an export company, the clothes are stored in boxes and are shipped in three different ships. Each box has a number on it. The boxes are stored in ships in specific pattern as below:

1. First box number in Ship1 + Last box number in Ship2 = first box number in Ship3.

2. Second box number in Ship1 + Second Last box number in Ship2 = Second box number in ship3.

Similarly all the boxes are numbered in ship3. Assuming the count of boxes in each ship is equal, find the box numbers in ship3.

Include a function named **arrayAddition** that accepts 3 arguments and its 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 is stored in an array variable named output1.

If the size of the array (the number of boxes) 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(i.e box numbers in ship 1). The last 'n' integers correspond to the elements in the second array (i.e box numbers in ship 2). If any of the inputs are invalid, then terminate the program.

Output consists of n integers that correspond to the box numbers in ship 3.

Assume that the maximum size of the array is 20.

**Sample Input 1 :**

5

23

2

5

32

76

2

9

21

42

4

**Sample Output 1 :**

27

44

26

41

78

**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 arrayAddition(int[],int[],int);

int main(){

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

}

}

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

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

if(input2[i]<0){

printf("Invalid Input");

getchar();

getchar();

exit(0);

}

}

arrayAddition(input1,input2,size);

return 0;

}

void arrayAddition(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();

}

**60.adjecentDifference**

Read the question carefully and follow the input and output format.  
  
Given an input Integer array, find the difference in the adjacent elements and print the largest difference  
  
**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 largest adjacent difference.  
  
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 adjecentDifference(int numbers[], int size) whose return type is an integer  
  
**Sample Input 1:**  
7  
2  
4  
5  
1  
9  
3  
8  
**Sample Output 1:**  
8  
  
Hint: The AdjecentElement Diff are :2,1,4,8,6,5 and Maximum diff is 8 which is obtained by 2-4 =2, 4-5=1,5- 1 =4, 1-9=8,9-3=6,3-8 =5  
  
**Sample Input 2:**  
5  
1  
7  
3  
-8  
**Sample Output 2:**  
Invalid input

#include<stdio.h>

#include<stdlib.h>

int output1[20];

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

int main(){

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

}

}

diff = adjecentDifference(input1,size);

printf("%d",diff);

getchar();

getchar();

return 0;

}

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

int i=0,j=0,max=0,diff=0;

for(i=0;i<size-1;i++){

if(numbers[i]>numbers[i+1])

diff=numbers[i]-numbers[i+1];

else

diff=numbers[i+1]-numbers[i];

if(diff>max)

max=diff;

}

return max;

}