**101.firstMiddleSame**

Read the question carefully and follow the input and output format.  
  
Write a program to check if the first and middle element in an array is the same, if so display “100” in output or else display the first element of the array.  
  
Note: Array size is always odd.  
  
**Input and Output Format :**  
The first line of the input consists of an integer, n that corresponds to the number of elements in the array.  
The next 'n' lines correspond to the elements in the 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.  
3) Print "Size must be odd" when the size of the array is even.  
  
Include a function named firstMiddleSame(int array[], int size) whose return type is an integer..  
  
**Sample Input 1:**  
7  
8  
4  
5  
8  
3  
2  
1  
  
**Sample Output 1:**  
100  
  
**Sample Input 2:**  
10  
4  
5  
6  
-8  
  
**Sample Output 2:**  
Invalid input  
  
**Sample Input 3:**  
3  
4  
8  
6  
  
**Sample Output 3:**  
4

#include<stdio.h>

#include<stdlib.h>

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

int main(){

int size=0,input1[20],i=0,res=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 id");

getchar();

getchar();

exit(0);

}

}

if(size%2==0){

printf("size must be odd");

getchar();

getchar();

exit(0);

}

res = firstMiddleSame(input1,size);

printf("%d",res);

getchar();

getchar();

return 0;

}

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

int mid=size/2,res=0;

if(array[0]==array[mid])

res = 100;

else

res = array[0];

return res;

}

**102.countEvenRepeat**

Read the question carefully and follow the input and output format.  
  
Write a program to count the number of repeated even elements in a given input array.  
  
**Input and Output Format :**  
First line of input consists of n, the number of elements. And the remaining n lines is the elements of the array.  
Output is a single integer that displays the count.  
  
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 countEvenRepeat(int array[], int size) whose return type is an integer, the count  
  
**Sample Input 1:**  
9  
8  
4  
5  
8  
4  
2  
1  
4  
5  
  
**Sample Output 1:**  
2  
  
**Sample Input 2:**  
10  
4  
5  
6  
-8  
  
**Sample Output 2:**  
Invalid input

#include<stdio.h>

#include<stdlib.h>

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

int main(){

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

}

}

res = countEvenRepeat(input1,size);

printf("%d",res);

getchar();

getchar();

return 0;

}

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

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

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

count=1;

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

if(array[i] == array[j]){

++count;

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

array[k] = array[k+1];

}

size--;

}

else

j++;

}

if(count!=1 && array[i]%2==0)

sum=sum+1;

}

return sum;

}

**103.Array Sorting**

Write a program to sort the first half of the input array elements in ascending order and the second half of the input array elements in descending order.

Include a function named **ascDescArray** that accepts 2 arguments and its return type is void. The first argument is the input array and the second argument is an int that corresponds to the size of the array.

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 n integers that correspond to the elements in the sorted array.

Assume that the maximum size of the array is 20.

**Sample Input 1:**

7

1

9

8

4

6

4

5

**Sample Output 1:**

1

4

8

9

6

5

4

**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>

void ascDescArray(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);

}

}

ascDescArray(input1,size);

getchar();

getchar();

return 0;

}

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

int i, j, k,size1=size/2+1,size2=size-size1;

int temp=0;

for(i=0;i<size1;i++){

for(j=i+1;j<size1;j++){

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

temp = array[i];

array[i] = array[j];

array[j] = temp;

}

}

}

for(i=size1;i<size;i++){

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

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

temp = array[i];

array[i] = array[j];

array[j] = temp;

}

}

}

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

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

}

**104.occurenceDigit**

Read the question carefully and follow the input and output format.  
  
Given a number n, count the occurences of a number,x in n.  
  
**Input and Output Format :**  
The first line of the input consists of an integer n, the second line consists of an integer, which is the digit whose occurence is to be calculated. Output is an integer that gives the count.  
  
Print "Number too small" if any of the 2 input numbers is less than 0 and terminate the program.  
Print "Number too large" if any of the 2 input numbers is greater than 32767 and terminate the program.  
  
Include a function named occurenceDigit(int number,int digit) that returns an integer, which is the count of the digit.  
  
**Sample Input 1:**  
1122  
1  
**Sample Output 1:**  
2  
  
  
**Sample Input 2:**  
-2134  
**Sample Output 2:**  
Number too small

#include<stdio.h>

#include<stdlib.h>

int occurenceDigit(int number,int digit);

int main(){

int number=0,result,find\_digit=0;

scanf("%d",&number);

scanf("%d",&find\_digit);

if(number<0 || find\_digit<0){

printf("Number too small");

getchar();

getchar();

exit(0);

}

else if(number>32767 || find\_digit>32767){

printf("Number too large");

getchar();

getchar();

exit(0);

}

else{

result = occurenceDigit(number,find\_digit);

printf("%d",result);

}

getchar();

getchar();

return 0;

}

int occurenceDigit(int number,int digit){

int rem,count=0;

while(number!=0){

rem=number%10;

if(rem==digit)

count++;

number=number/10;

}

return count;

}

**105.consecutiveDifference**

Read the question carefully and follow the input and output format.  
  
Given input as integer array in which consecutive elements should have a difference of 4 or more than 4.  
  
If the above condition matches display “1” else “0”  
  
**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 either 1 or 0  
  
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 consecutiveDifference(int elements[], int size) whose return type is an integer either 1 or 0.  
  
**Sample Input 1:**  
6  
1  
5  
10  
6  
2  
7  
  
**Sample Output 1:**  
1  
  
**Sample Input 2:**  
-8  
  
**Sample Output 2:**  
Invalid array size

#include<stdio.h>

#include<stdlib.h>

int consecutiveDifference(int elements[], int size);

int main(){

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

}

}

res = consecutiveDifference(input1,size);

if(res==1)

printf("%d",res);

else

printf("%d",res);

getchar();

getchar();

return 0;

}

int consecutiveDifference(int elements[], int size){

int i=0,j=0,res=0,diff=0,flag=0;

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

flag=0;

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

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

else

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

if(diff<4){

flag=1;

break;

}

}

if(flag==1)

res=0;

else

res=1;

return res;

}

**106.Change using Minimal Coins / Notes**

Ram needs to pay the school fees of his 6 year old kid. As he is busy with his work, he is not finding time to go to the school to make payment. His kid's school doesn't accept online payment. So he decided to send the fee amount through his kid. The available denominations of rupees or coins is 500, 100, 50, 10, 5 and 1. Can you write a program to find the minimal number of coins or notes to be given to his kid?

Include a function named **countRupees** that accepts an integer and returns an integer that corresponds to the minimal number of coins/rupee notes. The function returns -1 if the input is negative.

**Input and Output Format:**

Input consists of a single integer that corresponds to the fee amount to be paid.

Output consists of an integer that corresponds to the minimal number of coins or rupee notes.

Print Invalid Input if the input value is negative.

**Sample Input 1:**

682

**Sample Output 1:**

8

**Sample Input 2:**

**-**2345

**Sample Output 2:**

Invalid Input

#include<stdio.h>

#include<stdlib.h>

int countRupees(int);

int main(){

int fees,count=0;

scanf("%d",&fees);

count = countRupees(fees);

if(count==-1)

printf("invalid input");

else

printf("%d",count);

getchar();

getchar();

return 0;

}

int countRupees(int rupees){

int count=0;

int c1,c2,c3,c4,c5,c6;

if(rupees<0)

count=-1;

else

{

c1=rupees/500;

rupees=rupees-(500\*c1);

c2=rupees/100;

rupees=rupees-(100\*c2);

c3=rupees/50;

rupees=rupees-(50\*c3);

c4=rupees/10;

rupees=rupees-(10\*c4);

c5=rupees/5;

rupees=rupees-(5\*c5);

c6=rupees/1;

rupees=rupees-(1\*c6);

count=c1+c2+c3+c4+c5+c6;

}

return count;

}

**107.singleDoubleTripCount**

Read the question carefully and follow the input and output format.  
  
In a given input array find out number of single digits, double digits and triple digits.  
  
**Input and Output Format :**  
First line of input consists of an integer, the size of the array. Next line correspond to the elements of the array. Output consist of an integer 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 singleDoubleTripCount(int elements[], int size) whose return type is void.  
The output array is stored in a global variable named count. The seize of the output array is 6.  
**Sample Input 1:**  
5  
1  
2  
23  
34  
456  
  
**Sample Output 1:**  
1  
2  
2  
2  
3  
1  
  
**Sample Input 2:**  
4  
1  
2  
-3  
  
**Sample Output 2:**  
Invalid input

#include<stdio.h>

#include<stdlib.h>

int count[6];

void singleDoubleTripCount(int elements[], int size);

int main(){

int size=0,input1[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);

}

}

singleDoubleTripCount(input1,size);

getchar();

getchar();

return 0;

}

void singleDoubleTripCount(int elements[], int size){

int i=0,c1=0,c2=0,c3=0;

count[0]=1;

count[2]=2;

count[4]=3;

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

if(elements[i]>=0 && elements[i]<=9)

count[1]=++c1;

else if(elements[i]>=10 && elements[i]<=99)

count[3]=++c2;

else if(elements[i]>=100 && elements[i]<=999)

count[5]=++c3;

else

;

}

for(i=0;i<6;i++)

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

}

**108.Sum of multiples**

Given 2 integers n and m, write a program to find the sum of all numbers divisible by m upto and including n (i.e from 1 to n).

Include a function named **sumMultiples**that accepts 2 integer arguments and returns an integer that corresponds to the sum of the multiples. The first argument corresponds to n and the second argument corresponds to m. The function returns -1 if the any of the two inputs is less than 0.

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

**Input and Output Format:**

The input consists of 2 integers. The 1st integer corresponds to n and the 2nd integer corresponds to m.

The output consists of an integer that corresponds to the sum of the multiples.

**Sample Input 1:**

11

5

**Sample Ouput 1:**

15

**Sample Input 2:**

-8  
2

**Sample Output 2:**

Invalid Input

#include<stdio.h>

#include<stdlib.h>

int sumMultiples(int,int);

int main(){

int n=0,m=0,sum=0;

scanf("%d%d",&n,&m);

sum = sumMultiples(n,m);

if(sum == -1)

printf("Invalid Input");

else

printf("%d",sum);

getchar();

getchar();

return 0;

}

int sumMultiples(int n,int m){

int i,sum=0;

if(n<0 || m<0)

sum=-1;

else{

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

if(i%m==0)

sum = sum + i;

}

}

return sum;

}

**109.Search Element**

Write a program to find whether a particular number appears in a given input array.

Include a function named **isElementPresent** that accepts 3 arguments and returns an int. The first argument is the input array, the second argument is an int that corresponds to the size of the array and the third argument is the element to be searched for. The function returns 1 if the search element is present in the array and returns 0 if the search element is not present in the array.

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+2 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. The last integer corresponds to the element whose count needs to be found.

Output consists of a string that is either 'yes' or 'no'.

Refer sample output for formatting specifications.

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

**Sample Input 1:**

8

2

1

3

8

6

8

10

8

8

**Sample Output 1:**

yes

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

int main(){

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

}

}

scanf("%d",&element);

res = isElementPresent(input1,size,element);

if(res==1)

printf("yes");

else

printf("no");

getchar();

getchar();

return 0;

}

int isElementPresent(int numbers[], int size, int element){

int i=0,res=0;

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

if(numbers[i]==element){

res=1;

break;

}

}

return res;

}

**110.Binary Conversion**

Write a program to convert a given input decimal number to binary.

Include a function named **convertToBinary** that accepts an integer argument and its return type is void. Print the binary representation of the number in this function.

If the input value is negative or greater than 100, print Invalid Input and terminate the program.

**Input and Output Format:**

Input consists of a single integer.

Output consists of the binary representation of the given integer.

Refer sample output for formatting specifications.

**Sample Input 1:**

12

**Sample Output 1:**

1100

**Sample Input 2:**

64

**Sample Output 2:**

1000000

**Sample Input 3:**

**-**2345

**Sample Output 3:**

Invalid Input

#include<stdio.h>

#include<stdlib.h>

void convertToBinary(int num);

int main(){

int number=0;

scanf("%d",&number);

if(number<0 || number>100){

printf("Invalid input");

getchar();

getchar();

exit(0);

}

else

convertToBinary(number);

return 0;

}

void convertToBinary(int num){

int a[20],i=0;

while(num>0)

{

a[i]=num%2;

i++;

num=num/2;

}

for(int j=i-1;j>=0;j--)

printf("%d",a[j]);

getchar();

getchar();

}

**111.Repeated Element**

Write a program to find the maximum repeated element in a given input array.

Include a function named **maxRepeatedElement** 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 maximum repeated 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 maximum repeated element.

Assume that the maximum number of elements in the array is 20 and that there will always be a unique maximum repeated element.

**Sample Input 1:**

8

2

1

3

4

6

8

10

8

**Sample Output 1:**

8

**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 maxRepeatedElement(int array[], int size);

int main(){

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

}

}

res = maxRepeatedElement(input1,size);

printf("%d",res);

getchar();

getchar();

return 0;

}

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

int i, j, k=0;

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

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

if(array[i] == array[j]){

k=array[i];

break;

}

}

}

return k;

}

/\*

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

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

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

count=1;

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

if(array[i] == array[j]){

++count;

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

array[k] = array[k+1];

}

size--;

}

else

j++;

}

if(count!=1 && array[i]%2==0)

sum=sum+1;

}

return sum;

}

\*/

**112.subTwoArrays**

Read the question carefully and follow the input and output format.  
  
Given two input arrays,  write a program to find out numbers which is present in the first array and not in the second array.  
  
**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.  
  
1) Print "Invalid array size" when size of the array is a negative number.  
2) Print "Invalid input" when there is any negative numbers available in the input array.  
  
  
Include a function named subTwoArrays(int elements1[], int elements2[], int size) whose return type is void.  
The output array is stored in a global variable named no\_common.  
  
**Sample Input 1:**  
5  
1  
2  
3  
4  
5  
3  
5  
7  
9  
10  
  
**Sample Output 1:**  
1  
2  
4  
  
**Sample Input 2:**  
4  
1  
2  
-3  
  
**Sample Output 2:**  
Invalid input

#include<stdio.h>

#include<stdlib.h>

int no\_common[20];

void subTwoArrays(int elements1[], int elements2[], 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);

}

}

subTwoArrays(input1,input2,size);

return 0;

}

void subTwoArrays(int elements1[], int elements2[], int size){

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

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

flag=0;

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

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

flag=1;

}

if(flag==0){

no\_common[k]=elements1[i];

k++;

}

}

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

printf("%d\n",no\_common[i]);

getchar();

getchar();

}

**113.primeFactorialSum**

Read the question carefully and follow the input and output format.  
  
In a given input number , find out the sum of factorial of digits that are prime.  
  
**Input and Output Format :**  
Input consists of an integer. Output consists of the factorial sum.  
1) Print "Number too large" when the given input number is greater than 32767  
2) Print "Number too small" when the given input number is a negative number.  
  
Include a function named primeFactorialSum(int number) whose return type is an integer.  
  
**Sample Input 1:**  
123  
**Sample Output 1:**  
8  
  
Hint : 2! + 3! = (8)  
  
**Sample Input 2:**  
32768  
**Sample Output 2:**  
Number too large

#include<stdio.h>

#include<stdlib.h>

int primeFactorialSum(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 = primeFactorialSum(number);

printf("%d",result);

}

getchar();

getchar();

return 0;

}

int primeFactorialSum(int number){

int i,count=0,num,sum=0,rem=0,flag=0,k,res=1;

while(number!=0){

rem=number%10;

count=0;

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

if(rem%i==0)

count++;

}

if(count==2){

res=1;

for(k=1;k<=rem;k++)

res = res \* k;

sum = sum + res;

}

number=number/10;

}

return sum;

}