1.calculateSal

Read the question carefully and follow the input and output format.

Karen got salary for this month and she spends 20% of her salary for food and 30% of her salary for travel. If she takes care of other shifts she will get 2% of the salary per day. Given her salary and the number of shifts she handled. Calculate how much she can save in her pocket after spending all these?

Input and Output Format :

First line of input consists of an integer, salary. Next line correspond to the number of shifts. Output consist of an integer, which is saving.

1) Print "Salary too large" when salary is greater than 8000.
2) Print "Shifts too small" when the shift is less than 0.
3) Print "Salary too small" when the salary is less than 0.

Include a function named calculateSal(int salary, int shifts) whose return type is an integer, which is the saving.

Sample Input 1:

7000
5

Sample Output 1:

4200

Sample Input 2:

8001

Sample Output 2:

Salary too large

```
#include<stdio.h>
#include<stdlib.h>
int calculateSal(int, int);
int main(){
	int salary=0,shifts=0,savings=0;
	scanf("%d",&salary);
	scanf("%d",&shifts);
	if(salary>8000)
		printf("Salary too large\n");
	else if(shifts<0)
		printf("Shifts too small\n");
	else if(salary<0)
		printf("Salary too small");
	else{
		savings = calculateSal(salary,shifts);
		printf("%d",savings);
	}
	getchar();
	getchar();
	return 0;
}
int calculateSal(int salary, int shifts){
	int saving=0;
	saving = (salary*0.5)+(salary*0.02*shifts);
	return saving;
}
```

2.Repeated Salary Count

John is working as a clerk in an organization where N number of people are working. His boss has asked him to get the count of employees who get same salary. Help him to get the count of repeated salary.

Include a function named **countRepeaters** 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 number of repeaters.

If the size of the array is negative or if any of the array elements are 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 number of repeaters.

Assume that utmost one element in the array would repeat.
Assume that the maximum number of elements in the array is 20.

Sample Input 1:
5
1000
2000
3500
2000
5000

Sample Output 1:
2

Sample Input 2:
-5

Sample Output 2:
Invalid Input

Sample Input 3:
5
1000
-2000

Sample Output 3:
Invalid Input

```
#include<stdio.h>
#include<stdlib.h>
int countRepeaters(int[],int);
int main(){
	int n=0,input[20]={0},i,flag=0,No_of_elements=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);
	}
	No_of_elements = countRepeaters(input,n);
	printf("%d",No_of_elements);
	getchar();
	getchar();
	return 0;
}
int countRepeaters(int in[],int size){
	int elements_count=1,i,j,out[20],k=0;
	for(i=0;i<size;i++){
		for(j=i+1;j<size;){
			if(in[i]==in[j]){
				elements_count++;
				for(k=j;k<size;k++)
					in[k]=in[k+1];
				size--;
			}
			else
				j++;
		}
	}
	return elements_count;
}
```

3.maximumSum

Read the question carefully and follow the input and output format.

Given an Integer array, find out sum of Even and odd Numbers individually and find the maximum.

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 maximum of odd and even 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 maximumSum(int numbers[], int size) whose return type is an integer,.

Sample Input 1:
5
12
13
14
15
16

Sample Output 1:
42

Sample Input 2:
-13

Sample Output 2:
Invalid array size

```
#include<stdio.h>

#include<stdlib.h>

int maximumSum(int[],int);

int main(){

        int n=0,i,flag=0,input[20],max=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);
        }
        max = maximumSum(input,n);
        printf("%d",max);
        getchar();
        getchar();
        return 0;
}
int maximumSum(int numbers[], int size){
        int i=0,max=0,evensum=0,oddsum=0;
        for(i=0;i<size;i++){
                if(numbers[i]%2==0)
                        evensum = evensum + numbers[i];
                else
                        oddsum = oddsum + numbers[i];
```

```
            if(evensum>oddsum)
                max = evensum;
            else
                max=oddsum;
        }
        return max;
}
```

4.Product of Digits

In a car racing video game, the car is an object. You can drive the car, turn the car, or stop the car when needed but you need to drive long. You will get money according to the Km you have travelled. For example if you have travelled 123 km then the product of the km (ie 1*2*3 = 6) would be the amount you win. Write a program to find the product of the digits in the given input number.

Include a function named **productDigits** that accepts an integer argument and returns an integer that corresponds to the product of 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.

Input and Output Format:
Input consists of an integer.
Output consists of an integer.
Refer sample output for formatting specifications.

Sample Input 1:
32

Sample Output 1:
6

Sample Input 2:
-67

Sample Output 2:
Invalid Input

```
#include<stdio.h>
#include<stdlib.h>
```

```
int productDigits(int);
int main(){
	int n=0,result=0;
	scanf("%d",&n);
	result = productDigits(n);
	if(result==-1)
		printf("Invalid Input");
	else
		printf("%d",result);
	getchar();
	getchar();
	return 0;
}
int productDigits(int number){
	int num=1;
	signed int res=0;
	if((number<0) || (number>32767))
		res = -1;
	else{
		while(number!=0){
			num=num*(number%10);
			number = number/10;
		}
		res = num;
	}
	return res;
}
```

5.findCricketerId

Read the question carefully and follow the input and output format.

Given an input array first Index indicates the cricketer’s id and second index indicates the score and so on……Find out the cricketer's id who scored more than given score

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 of the input consists of an integer that corresponds to the given score. Output consist of an integer array, which contains cricketer's id who have scored more than the given score.

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.
3) Print "Invalid score" when the score is negative.

Include a function named findCricketerId(int array[], int size, int score) whose return type is void.
The output array is stored in a global variable named cricketer.

Sample Input 1:

6
1
1000
5
2000
3
4000
1000

Sample Output 1:

5
3

Sample Input 2:

6
1
1000
5
3000
3
4000
-1000

Sample Output 2:

Invalid score

```
#include<stdio.h>

#include<stdlib.h>
```

```
int cricketer[20];
void findCricketerId(int[], int, int);
int main(){
	int n=0,score=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);
	}
	scanf("%d",&score);
	if(score<0){
		printf("Invalid score");
		getchar();
		getchar();
		exit(0);
	}
```

```
	findCricketerId(input,n,score);
	getchar();
	return 0;
}
void findCricketerId(int array[], int size, int score){
	int i,j=0;
	for(i=1;i<size;i=i+1){
		if(array[i]>score){
			cricketer[j]=array[i-1];
			j++;
		}
	}
	for(i=0;i<j;i++){
		printf("%d\n",cricketer[i]);
	}
	getchar();
	getchar();
}
```

6.Fahrenhiet to Centigrade

Write a program to convert given temperature from Fahrenheit to Centigrade.

Formula:
C/5 = (F-32)/9
C stands for Centigrade.
F stands for Fahrenheit.

Include a function named **convertToCentigrade** that accepts an integer argument and returns a float that corresponds to the centigrade equivalent.

If the input is a negative number, print Invalid Input and terminate the program.

Input and Output Format:
Input consists of a single integer.

Output consists of a floating point number that corresponds to the centigrade equivalent. The output is displayed correct to 2 decimal places.

Sample Input 1:
77

Sample Output 1:
25.00

Sample Input 2:
-2345

Sample Output 2:
Invalid Input

```
 #include<stdio.h>

#include<stdlib.h>

float convertToCentigrade(int);

int main(){

        int f=0;

        float res=0.0f;

        scanf("%d",&f);

        if(f<0){

                printf("Invalid input");

                getchar();

                getchar();

                exit(0);

        }

        res = convertToCentigrade(f);

        printf("%0.2f",res);

        getchar();
```

```
	getchar();

	return 0;

}

float convertToCentigrade(int fah){

	float centigrade=0.0f;

	centigrade = float(fah-32)/9.0f*5;

	return centigrade;

}
```

7.highestFeedBack

Read the question carefully and follow the input and output format.

In a company there are some managers working on two different projects (MetLife and Hardfort). When the feedback was taken their feedback was present in both MetLife Feedback as well as Hardfort Feedback. Write a method to create a consolidated feedback for the managers for MetLife and HardForts. For those working on both the projects the highest feedback is taken. In the 2 given arrays, the First Index represents the Employee id and second one Represents The Feed Back Score and so on....

Input and Output Format:

First line corresponds to n, the size of the array. The next n lines correspond to the elements of the first array. The next n lines correspond to the elements in the second array. Output corresponds to the consolidated feedback score.

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 highestFeedBack(int metlife[],int hardfort[],int size) whose return type is void. The output array is stored in a global variable named fedback.

Sample Input 1:

```
8
1
90
2
75
3
92
5
```

85
1
80
2
85
3
80
4
85

Sample Output 1:

1
90
2
85
3
92
5
85
4
85

Sample Input 2:

5
5
8
9
1
-6

Sample Output 2:

Invalid number

Sample Input 3:

-4

Sample Output 3:

Invalid array size

```
#include<stdio.h>

#include<stdlib.h>

int fedback[30];

void highestFeedBack(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 array size");
		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);
	}
	highestFeedBack(input1,input2,size);

	return 0;
}
void highestFeedBack(int metlife[],int hardfort[],int size){
	int i=0,j=0,k=0,count=0,count1=0;
	for(i=0;i<size;i=i+2){
```

```
        count=0;
        for(j=0;j<size;j=j+2){
            if(metlife[i]==hardfort[j]){
                count=1;
                if(metlife[i+1]>hardfort[j+1]){
                    fedback[k]=metlife[i];
                    fedback[++k]=metlife[i+1];
                    k++;
                }
                else{
                    fedback[k]=metlife[i];
                    fedback[++k]=hardfort[j+1];
                    k++;
                }
            }
        }
        if(count==0){
            fedback[k]=metlife[i];
            fedback[++k]=metlife[i+1];
            k++;
        }
}
for(i=0;i<size;i=i+2){
        count1=0;
        for(j=0;j<size;j=j+2){
            if(hardfort[i]==metlife[j]){
                count1=1;
            }
        }
```

```
            if(count1!=1){
                fedback[k]=hardfort[i];
                fedback[++k]=hardfort[i+1];
                k++;
            }
        }
        for(i=0;i<k;i++){
            printf("%d\n",fedback[i]);
        }

        getchar();
        getchar();
}
```

8.primeIndexSum

Read the question carefully and follow the input and output format.

Given an Integer array. Find the average of the numbers located on the Prime Indexes of the Array. Consider 0 index as 1 and 1 index is 2 and so on……

Hint :Consider 1 is not a prime number

Input and Output Format :

First line of input consists of n, the number of elements. Next n lines correspond to the array elements . Output consists of an Integer, the prime index sum.

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 primeIndexSum(int array[], int size) whose return type is an integer,which is the sum.

Sample Input 1:

7
2
4
5
1
9
3

8

Sample Output 1:
6

Sample Input 2:
-7

Sample Output 2:
Invalid array size

```
#include<stdio.h>
#include<stdlib.h>
int primeIndexSum(int[], int);
int main(){
	int n=0,flag=0,i,input[30];
	int avg=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)
			flag=1;
	}
	if(flag==1){
		printf("Invalid Input");
		getchar();
```

```
            getchar();
            exit(0);
        }
        avg = primeIndexSum(input,n);
        printf("%d",avg);
        getchar();
        getchar();
        return 0;
}
int primeIndexSum(int array[], int size){
        int sum=0,i,j=0,count=0,temp=0,avg=0;
        for(i=2;i<=size;i++){
            count=0;
            for(j=1;j<=i;j++){
                if(i%j==0)
                    count++;
            }
            if(count==2){
                sum = sum+array[i];
                temp++;
            }
            else
                continue;
        }
        avg=sum/temp;
        return avg;
}
```

9.Element Count

Write a program to find the number of times a particular number occurs in a given input array.

Include a function named **findElementCount** 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 an int that corresponds to the number of times the search element occurs 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 an integer that corresponds to the number of times the search element occurs in the array.
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:
3

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 findElementCount(int,int[],int);
int main(){
	int n=0,flag=0,i,input[20],search=0;
	int count=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);
        }
        scanf("%d",&search);
        count = findElementCount(n,input,search);
        printf("%d",count);
        getchar();
        getchar();
        return 0;
}
int findElementCount(int n,int array[],int find){
        int count=0,i;
        for(i=0;i<n;i++){
                if(array[i]==find)
                        count++;
        }
        return count;
}
```

10.powerOfTwo

Read the question carefully and follow the input and output format.

Check whether given number is a power of 2 or not .If yes Print 'Yes' else 'No'

Input and Output Format :
Input consists of an integer number. And output is a single line that displays 'Yes' or 'No'

Print "Number too small" if the number is less than 0
Print "Number too large" if the number is greater than 32767

Include a function named powerOfTwo(int n) that returns an integer.

Sample Input 1:
3
Sample Output 1:
No

Sample Input 2:
34569
Sample Output 2:
Number too large

```
#include<stdio.h>

#include<stdlib.h>

int powerOfTwo(int n);

int main(){

	int number=0,result;

	scanf("%d",&number);

	if(number<0)

		printf("Number too small");

	else if(number>32767)

		printf("Number too large");

	else{

		result=powerOfTwo(number);

		if(result==1)

			printf("yes");

		else

			printf("No");

	}

	getchar();

	getchar();
```

```
        return 0;
}
int powerOfTwo(int n){
        int result=0;
        while (((n % 2) == 0) && n > 1)
        n /= 2;
return (n == 1);
}
```

11.Count of 3 Multiples

Write a program to find the count of 3 multiples in a given input integer array.

Include a function named **divisibleBy3** 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 count of 3 multiples.

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 count of 3 multiples

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

Sample Input 1:
8
1
6
3
5
61
80
102
9

Sample Output 1:
4

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 divisibleBy3(int[],int);
int main(){
	int n=0,i,flag=0,input[20],count=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);
        }
        count = divisibleBy3(input,n);
        printf("%d",count);
        getchar();
        getchar();
        return 0;
}
int divisibleBy3(int array[],int size){
        int count=0,i;
        for(i=0;i<size;i++){
                if(array[i]%3==0)
                        count++;
        }
        return count;
}
```

12.Odd Even Average

The Owner of a block visited the Layout and found that he has some plot numbers of his own and some are odd numbers and some are even numbers. He is maintaining the details in a file in the system. For the password protection our owner has followed one formula. He calculated the sum of his even numbers plot and sum of odd numbers plot and found the average of those two and he used that average as his password for the details file. Find the password that our owner has arrived.

Include a function named **avgOddEvenSum** that accepts 2 arguments and returns a float. 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 a float that corresponds to the average 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 a floating point number that corresponds to the average. It is displayed correct to 2 decimal places.
Assume that the maximum size of the array is 20.

Sample Input 1:
5
1
2
3
4
5

Sample Output 1:
7.50

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>

float avgOddEvenSum(int[],int);

int main(){

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

        float avg=0.0f;

        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);
	}
	avg = avgOddEvenSum(input,n);
	printf("%0.2f",avg);
	getchar();
	getchar();
	return 0;
}
float avgOddEvenSum(int array[],int size){
	float avg;
	int i,sumodd=0,sumeven=0;
	for(i=0;i<size;i++){
```

```
            if(array[i]%2==0)
                    sumeven=sumeven+array[i];
            else
                    sumodd=sumodd+array[i];
    }
    avg = float(sumeven+sumodd)/2.0f;
    return avg;
}
```

13.Decimal Conversion

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

Include a function named **convertToDecimal** that accepts an integer argument and returns an integer that corresponds to the decimal representation of the input number. If the input value is not a binary value or if the input is negative or if the input is greater than 11111, the function returns -1.

If the function returns -1, print Invalid Input.

Input and Output Format:
Input consists of a single integer that corresponds to the binary representation of a number.
Output consists of a single integer that corresponds to the decimal equivalent of the given number.
Refer sample output for formatting specifications.

Sample Input 1:
1100

Sample Output 1:
12

Sample Input 2:
101010

Sample Output 2:
Invalid Input

Sample Input 3:
1201

Sample Output 3:
Invalid Input

```
#include<stdio.h>

#include<stdlib.h>

#include<string>

int convertToDecimal(int);

int findbinary(int);

int main(){

        int num=0,res=0;

        scanf("%d",&num);

        res = convertToDecimal(num);

        if(res==-1)

                printf("Invalid input");

        else

                printf("%d",res);

        getchar();

        getchar();

        return 0;

}

int convertToDecimal(int number){

        int decimal=0,res,j=1,rem;

        res = findbinary(number);

        if(res==0 || number<0 || number>11111)

                decimal = -1;

        else{

                while(number!=0){

    rem=number%10;

    decimal=decimal+rem*j;
```

```
        j=j*2;
        number=number/10;
    }
            }
            return decimal;
}

int findbinary(int number){
            int dv;
            while(number!=0)
            {
                        dv=number%10;
                        if(dv>1)
                                    return 0;
                        number=number/10;
            }
            return 1;
}
```

14.Arithmetic Operation

Write a program to perform a specific arithmetic operation

Include a function named **performArithmeticOperation** that accepts 3 integer arguments and returns an integer that corresponds to the result. The first and second arguments correspond to the input numbers and the third argument corresponds to the choice of arithmetic operation.

If argument 3 =1, calculate the sum of input1 and input2
If argument 3 =2, calculate the difference of input1 and input2
If argument 3 =3, calculate the product of input1 and input2
If argument 3 =4, calculate the quotient of input1 divided by input 2

If the first two argument's values is negative or greater than 32767 , the function returns -1.

If the third argument's value is not in the range 1 to 4, the function returns -1.

If the function returns -1, print Invalid Input.

Input and Output Format:
Input consists of 3 integers.
Output consists of an integer.
Refer sample output for formatting specifications.

Sample Input 1:
4
12
3

Sample Output 1:
48

Sample Input 2:
-67
2
1

Sample Output 2:
Invalid Input

```
 #include<stdio.h>

#include<stdlib.h>

#include<string>

int performArithmeticOperation(int,int,int);

int main(){

        int a,b,c,res=0;

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

        res=performArithmeticOperation(a,b,c);

        if(res==-1)

                printf("Invalid input");

        else
```

```
            printf("%d",res);
        getchar();
        getchar();
        return 0;
}
int performArithmeticOperation(int a,int b,int choice){
        int result=0;
        if(a<0 || b<0 || a>32767 || b>32767)
            result = -1;
        else if((choice<1)||(choice>4))
            result = -1;
        else
        {
            switch(choice){
            case 1:
                result = a+b;
                break;
            case 2:
                result = a-b;
                break;
            case 3:
                result = a*b;
                break;
            case 4:
```

```
                result = a/b;

                break;

            }

        }

        return result;

}
```

15.digitFactorial

Read the question carefully and follow the input and output format.

In a given input number , find out the factorial of each individual digit and assign it to output array.

Input and Output Format:
Input consists of a single integer. Output consists of an Integer array, the individual factorials.

Print "Number too large" when the given input numbers is greater than 32767 .
Print "Number too small" when the given input is a negative number.

Include a function named digitFactorial(int number) whose return type is void.
The output array is stored in a global variable named factorial.

Sample Input 1:
123

Sample Output 1:
1
2
6

Sample Input 2:
-2526

Sample Output 2:
Number too small

```
#include<stdio.h>

#include<stdlib.h>

void digitFactorial(int number);
```

```
int factorial[20];
int main(){
	int num=0;
	scanf("%d",&num);
	if(num>32767)
		printf("Number too large");
	else if(num<0)
		printf("Number too small");
	else{
		digitFactorial(num);
	}
	getchar();
	getchar();
	return 0;
}
void digitFactorial(int number){
	int n,i,fact,k=0;
	while(number!=0){
		n=number%10;
		fact=1;
		for(i=1;i<=n;i++)
			fact = fact * i;
		factorial[k]=fact;
		k++;
		number=number/10;
	}
	for(i=k-1;i>=0;i--)
		printf("%d\n",factorial[i]);
}
```

16.searchKeys

Read the question carefully and follow the input and output format.

Given an integer array, first index represents the key & second index represents the value. Find keys for the given value.

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 consistts of an integer that represents the value to be searched.
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.
3) Print "Key not found" when there is no keys found.

Include a function named searchKeys(int array[], int size) whose return type is void.
The output array is stored in a global variable named found.

Sample Input 1:

8
1
4
2
4
3
4
5
6
4

Sample Output 1:

1
2
3

Sample Input 2:

5
5
6
7
8
9
-5

Sample Output 2:
Key not found

```
#include<stdio.h>

#include<stdlib.h>

void searchKeys(int array[], int size,int search);

int found[20];

int main(){

        int n,input[20],search;

        int 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);

        }

        scanf("%d",&search);

        searchKeys(input,n,search);
```

```
	return 0;
}
void searchKeys(int array[], int size,int search){
	int i,k=0,flag=0;
	for(i=1;i<size;i=i+1){
		if(array[i]==search){
			flag=1;
			found[k]=array[i-1];
			k++;
		}
	}
	if(flag==0)
		printf("Key not found");
	else
	{
		for(i=0;i<k;i++)
			printf("%d\n",found[i]);
	}
	getchar();
	getchar();
}
```

17.passCount

Read the question carefully and follow the input and output format.

Given a input array, First index Represents RollNo second index represents Mark and so on. Write a program to find the number of students who had cleared the exam.

Note : If marks >=70 then He /she Cleared the exam. Array size is always even.

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,

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 passCount(int array[], int size) whose return type is an integer , the count.

Sample Input 1:
8
1
70
2
55
3
75
4
80

Sample Output 1:
3

Sample Input 2:
5
6
2
8
-2
Sample Output 2:
Invalid input

```
#include<stdio.h>

#include<stdlib.h>

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

int found[20];

int main(){

        int n,input[20],count;

        int 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);
        }
        count = passCount(input,n);
        printf("%d",count);
        getchar();
        getchar();
        return 0;
}
int passCount(int array[], int size){
        int count=0,i;
        for(i=1;i<size;i=i+1){
                if(array[i]>=70)
                        count++;
        }
        return count;
```

}

18.5 Multiples --- Average

Write a program to find the average of multiples of 5 upto 'n'. n is given as input.

Include a function named **findAverageBy5s** that accepts an integer argument and returns a float that corresponds to the average of multiples of 5.

If the input value is negative or greater than 32767, print Invalid Input and terminate the program.

Input and Output Format:
Input consists of a single integer.
Output consists of a floating point number. Output is displayed correct to 2 decimal places.
Refer sample output for formatting specifications.

Sample Input 1:
10

Sample Output 1:
7.50

Sample Input 2:
-67

Sample Output 2:
Invalid Input

```
#include<stdio.h>

#include<stdlib.h>

float findAverageBy5s(int);

int main(){

        int num=0;

        float avg;

        scanf("%d",&num);

        if(num<0 || num>32767){
```

```
                printf("Invalid input");
                exit(0);
        }
        avg = findAverageBy5s(num);
        printf("%0.2f",avg);
        getchar();
        getchar();
        return 0;
}
float findAverageBy5s(int number){
        float avg=0.0f;
        int i,sum=0,count=0;
        for(i=5;i<=number;i++){
                if(i%5==0){
                        sum=sum+i;
                        count++;
                }
        }
        avg = (float)sum/count;
        return avg;
}
```

19.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;
}
```

20.Largest Array

Write a program which takes two arrays of the same size as a input and compares the first element of first array with the first element of second array and stores the largest of these into the first element of the output array. Repeat the process till the last element of the first array is checked with the last element of the second array.

Include a function named **largestArray** that accepts 3 arguments and its return type is void. The first argument is input array 1, the second argument is 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 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 the largest array.
Assume that the maximum number of elements in the array is 20.

Sample Input 1:
4
2
1
3
4
1
9
2
8

Sample Output 1:
2
9
3
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>

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

int output1[20];

int main(){

        int n=0,flag=0,i,input1[20],input2[20];
```

```
	int count=0,flag1=0;
	scanf("%d",&n);
	if(n<0){
		printf("Invalid Input");
		getchar();
		getchar();
		exit(0);
	}
	for(i=0;i<n;i++){
		scanf("%d",&input1[i]);
		if(input1[i]<0)
			flag=1;
	}
	for(i=0;i<n;i++){
		scanf("%d",&input2[i]);
		if(input2[i]<0)
			flag1=1;
	}
	if(flag==1 || flag1==1){
		printf("Invalid Input");
		getchar();
		getchar();
		exit(0);
	}
	largestArray(input1,input2,n);
	return 0;
}
```

```
void largestArray(int array1[],int array2[],int n){
	int i;
	for(i=0;i<n;i++){
		if(array1[i]>array2[i]){
			output1[i]=array1[i];
		}
		else{
			output1[i]=array2[i];
		}
	}
	for(i=0;i<n;i++){
		printf("%d\n",output1[i]);
	}
	getchar();
	getchar();
}
```

21.Sum of Prime Numbers

Write a program to find the sum of the prime numbers present in the given input array.

Include a function named **sumPrime** that accepts 2 arguments and returns an int. The first argument is a pointer to the input array and the second argument is an int that corresponds to the size of the array. The function returns the sum of the prime numbers in the input 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.

Please note that 1 is neither prime nor composite.

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.

Refer sample output for formatting specifications.

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

Sample Input 1:
5
2
4
8
9
11

Sample Output 1:
13

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 sumPrime(int*,int);

int output1[20];

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 = sumPrime(input,n);
        printf("%d",sum);
        getchar();
        getchar();
        return 0;
}
int sumPrime(int *a,int n){
```

```
    int sum=0,i=0,j,count=0;
    for(i=0;i<n;i++,a++){
        count=0;
        for(j=1;j<=*a;j++){
            if((*a)%j==0)
                count++;
        }
        if(count==2)
            sum=sum+(*a);
    }
    return sum;
}
```

22.Leap Year

Write a program to find whether the given input year is a Leap Year.

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

1. 1 if the input is a Leap Year
2. 0 if the input is not a Leap Year
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:
2000

Sample Output 1:
yes

Sample Input 2:
1610

Sample Output 2:
no

Sample Input 3:
-2345

Sample Output 3:
Invalid Input

```
#include<stdio.h>
#include<stdlib.h>
int checkLeapYear(int);
int main(){
	int year=0,status=0;
	scanf("%d",&year);
	status = checkLeapYear(year);
	if(status==-1){
		printf("Invalid input");
	}
	else if(status==1){
		printf("yes");
	}
	else{
		printf("no");
	}
	getchar();
	getchar();
```

```
	return 0;
}
int checkLeapYear(int year){
	int result;
	if(year<0)
		result=-1;
	else if(year%4==0)
		result=1;
	else
		result=0;
	return result;
}
```

23.secondLargest

Read the question carefully and follow the input and output format.

Write a function to find second largest number in the given input integer array.

Assume that all elements in the input array are unique.

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 second largest.

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 secondLargest(int array[], int size) whose return type is an integer, the second largest.

Sample Input 1:

```
5
3
342
53
2
12
```

Sample Output 1:

53

Sample Input 2:
5
3
342
53
-2

Sample Output 2:
Invalid input

```
#include<stdio.h>
#include<stdlib.h>
int cricketer[20];
int secondLargest(int array[], int size);
int main(){
	int n=0,largest=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);
        }
        largest = secondLargest(input,n);
        printf("%d",largest);
        getchar();
        getchar();
        return 0;
}
int secondLargest(int array[], int size){
        int largest=0,i,j,b;
        for(i=0 ; i<size ; i++)
        {
                for(j=i ; j<size ; j++)
                {
                        if(array[i] < array[j])
                        {
                                b = array[j] ;
                                array[j] = array[i] ;
                                array[i] = b ;
                        }
                }
        }
        return array[1];
}
```

24.findFirstLargest

Read the question carefully and follow the input and output format.

Write a function to find the product of the first and the third largest numbers in the given input integer

array.

Input and Output Format :
First line of input corresponds to n, the size of array and next n lines correspond to the elements of the array

Print "Invalid array size" when size of the array is a negative number and terminate the program
Print "Invalid number" when there is any negative numbers available in the input array and terminate the program.

Include a function named findFirstLargest(int n, int array[]) that returns an integer, product of first and third largest numbers.

Sample Input 1:
5
11
241
83
8
2

Sample Output 1:
2651

Sample Input 2:
3
83
-8
Sample Output 2:
Invalid number

```
#include<stdio.h>

#include<stdlib.h>

int cricketer[20];

int findFirstLargest(int n, int array[]);

int main(){

        int n=0,largest=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 Number");
                getchar();
                getchar();
                exit(0);
        }
        largest = findFirstLargest(n,input);
        printf("%d",largest);
        getchar();
        getchar();
        return 0;
}
int findFirstLargest(int n,int array[]){
        int largest=0,i,j,b,k;
        /*To remove duplicates*/
        for(i=0 ; i<n ; i++){
                for(j=i+1 ; j<n ;){
                        if(array[i]==array[j]){
                                for(k=j;k<n;k++)
                                        array[k]=array[k+1];
                                n--;
                        }
```

```
                else
                        j++;
        }
    }
    /*to sort array in desc order*/
    for(i=0 ; i<n ; i++)
    {
        for(j=i ; j<n ; j++)
        {
            if(array[i] < array[j])
            {
                b = array[j] ;
                array[j] = array[i] ;
                array[i] = b ;
            }
        }
    }
    return (array[0] * array[2]);
}
```

25.generateCode

Read the question carefully and follow the input and output format.

In a game show everybody got one coupon with some code. They need to generate a code with only even numbers in that coupon. Find the answer.

Input and Output Format :

Input consists of an integer. Output consist of an integer, which is the generated code.

1) Print "Number too small" when the given input number is a negative number.
2) Print "Number too large" when the given input number is greater than 32767.
3) Print 0 If the coupon does not contain any even numbers.

Include a function named generateCode(int coupon) whose return type is an integer, which is the generated code.

Sample Input 1:

4352

Sample Output 1:
42

Sample Input 2:
1357

Sample Output 2:
0

Sample Input 3:
-1357

Sample Output 3:
Number too small

```
#include<stdio.h>
#include<stdlib.h>
int generateCode(int coupon);
int main(){
	int coupon=0,code=0;
	scanf("%d",&coupon);
	if(coupon<0)
		printf("Number too small");
	else if(coupon>32767)
		printf("Number too large");
	else{
		code = generateCode(coupon);
		if(code==0)
			printf("%d",0);
		else
			printf("%d",code);
	}
```

```
	getchar();
	getchar();
	return 0;
}
int generateCode(int coupon){
	int res=0,rem=0,i=1;
	while(coupon!=0){
		rem=coupon%10;
		if(rem%2==0){
			res=res+(rem*i);
			i=i*10;
		}
		coupon = coupon/10;
	}
	return res;
}
```

26.calculateBonus

Read the question carefully and follow the input and output format.

Given the basic salary as input, write a program to calculate the bonus and display it.

The bonus will be calculated based on the below category.
Basic>20000 bonus=17%of basic+1500
Basic>15000 bonus=15%of basic+1200
Basic>10000 bonus=12%of basic+1000
for rest =8%of basic+500

Input and Output Format :

First line of input consists of n, the basic salary.
Output is a single integer that displays the bonus.

Print "Number too large" when the given input numbers is greater than 32767 .
Print "Number too small" when the given input is a negative number.

Include a function named calculateBonus(int basic) whose return type is an integer, the bonus.

Sample Input 1:

21000

Sample Output 1:
5070

Sample Input 2:
327678
Sample Output 2:
Number too large

```
#include<stdio.h>
#include<stdlib.h>
int calculateBonus(int basic);
int main(){
	int salary=0,bonus=0;
	scanf("%d",&salary);
	if(salary<0)
		printf("Number too small");
	else if(salary>32767)
		printf("Number too large");
	else{
		bonus = calculateBonus(salary);
		printf("%d",bonus);
	}
	getchar();
	getchar();
	return 0;
}
int calculateBonus(int basic){
	int res=0;
	if(basic>20000)
		res=(basic*17/100)+1500;
	else if(basic>15000)
```

```
        res=(basic*15/100)+1200;
    else if(basic>10000)
        res=(basic*12/100)+1000;
    else
        res=(basic*8/100)+500;
    return res;
}
```

27.Minimum of 3

Write a program to find the minimum of 3 numbers.

Include a function named **findSmallest** that accepts 3 integer arguments and returns an integer that corresponds to the minimum value.

Input and Output Format:
Input consists of 3 integers.
Output consists of an integer.
Refer sample output for formatting specifications.

Sample Input 1:
4
12
3

Sample Output 1:
3

```
#include<stdio.h>
#include<stdlib.h>
int findSmallest(int,int,int);
int main(){
    int a,b,c,min=0;
    scanf("%d%d%d",&a,&b,&c);
    min = findSmallest(a,b,c);
    printf("%d",min);
```

```
	getchar();
	getchar();
	return 0;
}
int findSmallest(int a,int b,int c){
	int res=0;
	if(a<b && a<c)
		res=a;
	else if(b<a && b<c)
		res=b;
	else
		res=c;
	return res;
}
```

28.Sort and Delete

Write a program to delete the given number in the input array and then to sort the array.

Include a function named **sortAndDelete** that accepts 3 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 and the third argument is the array element to be deleted. The number of elements in the modified array is stored in the global variable named output1.

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

Please note that the elements in the array may not be unique.

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 to be deleted.
Output consists of an integer array.
Refer sample output for formatting specifications.

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

Sample Input 1:

8
1
6
3
5
8
10
4
8
8

Sample Output 1:

1
3
4
5
6
10

Sample Input 2:

-5

Sample Output 2:

Invalid Input

```
#include<stdio.h>
#include<stdlib.h>
void sortAndDelete(int[],int,int);
int output1[20];
int main(){
	int n=0,input[20]={0},i,flag=0,element=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);
	}
	scanf("%d",&element);
	sortAndDelete(input,n,element);
	return 0;
}
void sortAndDelete(int array[],int size,int element){
	int i,j,k=0,temp=0;
	for(i=0;i<size;i++){
		if(array[i]!=element){
			output1[k]=array[i];
			k++;
		}
	}
	for(i=0;i<k;i++){
		for(j=i;j<k;j++){
			if(output1[i]>output1[j]){
				temp = output1[j];
				output1[j] = output1[i];
				output1[i] = temp;
			}
		}
	}
	for(i=0;i<k;i++){
		printf("%d\n",output1[i]);
	}
	getchar();
	getchar();
}
```

29.generateNewNumber

Read the question carefully and follow the input and output format.

Write a program to generate new number from the given input based on following conditions.

(i) Even digit should be replaced by next Even digit.
(ii) Odd digit should be replaced with next Odd digit

Input and Output Format :
Input consists of an integer. Output is also an integer.

1) Print "Number too small" when any of given input numbers is a negative number.

2) Print "Number too large" when any of given input numbers is greater than 32767.

Include a function named generateNewNumber(int number) whose return type is an integer, which is the replaced number.

Sample Input 1:
123

Sample Output 1:
345

Sample Input 2:
32768

Sample Output 2:
Number too large

```
#include<stdio.h>

#include<stdlib.h>

int generateNewNumber(int number);

int main(){

	int number=0,new_number=0;

	scanf("%d",&number);

	if(number<0)

		printf("Number too small");

	else if(number>32767)

		printf("Number too large");

	else{

		new_number = generateNewNumber(number);

		printf("%d",new_number);

	}

	getchar();

	getchar();

	return 0;

}

int generateNewNumber(int number){
```

```
    int res=0,rem=0,i=1;
    while(number!=0){
        rem=number%10;
        if(rem%2==0){
            res=(rem+2)*i+res;
            i=i*10;
        }
        else
        {
            res=(rem+2)*i+res;
            i=i*10;
        }
        number = number/10;
    }
    return res;
}
```

30.findMileage

Read the question carefully and follow the input and output format.

Given the cubic capacity(CC) of a bike. Write a function to return the mileage/liter for the given Cubic Capacity(CC). The mileage will be calculated as follows:

if CC is between 100 and 125, mileage is 75
if CC is between 126 and 135, mileage is 70
if CC is between 136 and 150, mileage is 60
if CC is between 151 and 200, mileage is 50
if CC is between 201 and 220, mileage is 35

First line of input consists of an integer that corresponds to CC of a bike. Output consist of an integer, which is the mileage.

Print "Number too large" when the given input CC is greater than 220.
Print "Number too small" when the given input CC is less than 100.

Include a function named findMileage(int cc) whose return type is an integer, which is the mileage.

Sample Input 1:

1

Sample Output 1:
Number too small

Sample Input 2:
160

Sample Output 2:
50

```
#include<stdio.h>

#include<stdlib.h>

int findMileage(int cc);

int main(){

	int cc=0,mileage=0;

	scanf("%d",&cc);

	if(cc<100)

		printf("Number too small");

	else if(cc>220)

		printf("Number too large");

	else{

		mileage = findMileage(cc);

		printf("%d",mileage);

	}

	getchar();

	getchar();

	return 0;

}

int findMileage(int cc){

	int mil=0;

	if(cc>=100 && cc<=125)

		mil=75;
```

```
        else if(cc>=126 && cc<=135)
                mil=70;
        else if(cc>=136 && cc<=150)
                mil=60;
        else if(cc>=151 && cc<=200)
                mil=50;
        else if(cc>=201 && cc<=200)
                mil=35;
        else
                ;
        return mil;
}
```

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();
}
```

41.University Type

Write a program to find if the student is eligible for first, second or third grade universities by finding the average of their marks given in the input integer array.

Grade should be calculated as given below :
Average >80 First Grade University
Average >60 Second Grade University
Otherwise Third Grade University

Include a function named **calculateGrade** that accepts 2 arguments and returns an integer. 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 integer that corresponds to the university type. The function returns 1 if the student is eligible for First Grade univesity, returns 2 if the student is eligible for Second Grade University, returns 3 if the student is eligible for Third Grade University and returns -1 if the average is greater than 99.

If the size of the array is negative or if any element in the array is negative or if the average marks scored by the student is greater than 99 , 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 a string --- “First Grade University” or “Second Grade University” or “Third Grade University” or “Invalid Input”.
Assume that the maximum size of the array is 20.

Sample Input 1:
5
92
87
78
74
80

Sample Output 1:
First Grade University

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 calculateGrade(int[],int);

int main(){

        int n=0,input[30],i,flag=0,grade=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);
}
grade=calculateGrade(input,n);
if(grade==-1)
            printf("Invalid Input");
else if(grade==1)
            printf("First Grade University");
else if(grade==2)
            printf("Second Grade University");
else if(grade==3)
            printf("Third Grade University");
else
            ;
getchar();
getchar();
```

```
	return 0;
}


int calculateGrade(int array[],int n){
	int grade=0,i,sum=0,avg=0;
	for(i=0;i<n;i++)
		sum=sum+array[i];
	avg=sum/n;
	if(avg>99)
		grade=-1;
	else if(avg>80)
		grade=1;
	else if(avg>60)
		grade=2;
	else
		grade=3;
	return grade;
}
```

42.Interchange Array

Write a program to interchange the first element in the array with the last element in the array. Repeat the process till the middle of the array.

Include a function named **interchangeArray** 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 the interchanged array.

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

Sample Input 1:
4
2
1
3
4

Sample Output 1:
4
3
1
2

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>
void interchangeArray(int[],int);
int main(){
	int n=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);
        }
        interchangeArray(input,n);
        return 0;
}

void interchangeArray(int array[],int n){
        int i=0,temp=0,j;
        for(i=0,j=n-1;i<n/2;i++,j--){
                temp = array[i];
                array[i] = array[j];
                array[j] = temp;
        }
        for(i=0;i<n;i++)
                printf("%d\n",array[i]);
```

```
    getchar();

    getchar();

}
```

43.Armstrong Number

Write a program to find whether the given input number is an Armstrong Number.

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

1. yes if the input is an Armstrong number
2. no if the input is not an Arnstrong number
3. Invalid Input if the input is a negative number or if the input is not a 3-digit 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:
153

Sample Output 1:
yes

Sample Input 2:
161

Sample Output 2:
no

Sample Input 3:
2345

Sample Output 3:
Invalid Input

```
#include<stdio.h>

#include<stdlib.h>

int checkArmstrong(int);

int main(){
```

```
	int number=0,result;
	scanf("%d",&number);
	result = checkArmstrong(number);
	if(result==1)
		printf("yes");
	else if(result==0)
		printf("no");
	else
		printf("Invalid input");
	getchar();
	getchar();
	return 0;
}

int checkArmstrong(int n){
	int res=0,temp=0,rem=0,sum=0;
	if(n<0 || n>999)
		res=-1;
	else{
		 temp = n;
		 while (temp != 0) {
			rem = temp%10;
			sum = sum + (rem*rem*rem);
			temp = temp/10;
		 }
		 if (n == sum)
			res=1;
```

```
            else
                res=0;
        }
        return res;
}
```

44.Factorial

Write a program to find the factorial of a given number.

Include a function named **findFactorial** that accepts an integer argument and returns an integer that corresponds to factorial. If the input value is negative or greater than 10, the function returns -1.

If the function returns -1, print Invalid Input.

Input and Output Format:
Input consists of a single integer.
Output consists of an integer.
Refer sample output for formatting specifications.

Sample Input 1:
4

Sample Output 1:
24

Sample Input 2:
-67

Sample Output 2:
Invalid Input

```
#include<stdio.h>
#include<stdlib.h>
int findFactorial(int);
int main(){
```

```
	int number=0,result;
	scanf("%d",&number);
	result = findFactorial(number);
	if(result==-1)
		printf("Invalid input");
	else
		printf("%d",result);
	getchar();
	getchar();
	return 0;
}

int findFactorial(int n){
	int res=1,i;
	if(n<0 && n>10)
		res=-1;
	else{
		for(i=1;i<=n;i++)
			res = res * i;
	}
	return res;
}
```

45.Salary Calculation

Jim got his salary. His salary calculations are as follows.
From his Basic amount he gets 50% of his basic for house Rent allowances and 75% of his basic as special allowances . If the number of days he worked is 31 he gets 500 extra. Write a program to calculate his gross salary after calculating all his salary split up.

Include a function named **calculateGross** that accepts 2 integer arguments and returns a float. The first integer corresponds to Jim's basic salary and the second integer corresponds to the number of days Jim has worked. The function returns a float that corresponds to the gross salary.

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

1. Basic salary is greater than 10000
2. Number of working days is greater than 31
3. Basic salary is negative
4. Number of working days is 0 or negative

Input and Output Format:
Input consists of 2 integers. The first integer corresponds to Jim's basic salary and the second integer corresponds to the number of days he has worked.
Output consists of a single float that corresponds to Jim's gross salary. The gross salary is displayed correct to 2 decimal places.

Sample Input 1:
5000
30

Sample Output 1:
11250 .00

Sample Input 2:
5000
0

Sample Output 2:
Invalid Input

```
#include<stdio.h>

#include<stdlib.h>

float calculateGross(int,int);

int main(){

	int basic=0,days=0;

	float gross=0.0f;

	scanf("%d%d",&basic,&days);

	if(basic>10000 || days>31 || basic<0 || days==0 || days<0){

		printf("Invalid input");

		getchar();

		getchar();
```

```
            exit(0);
        }
        gross = calculateGross(basic,days);
        printf("%0.2f",gross);
        getchar();
        getchar();
        return 0;
}

float calculateGross(int basic,int days){
        float salary=0.0f;
        if(days==31)
            salary=(float)(basic*50)/100+(float)(basic*75)/100+basic+500;
        else
            salary=(float)(basic*50)/100+(float)(basic*75)/100+basic;
        return salary;
}
```

46.Perfect Square

Write a program to find whether the given input number is a perfect square without using sqrt function.

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

1. 1 if the input is a perfect square
2. 0 if the input is not a perfect square
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:
36

Sample Output 1:
yes

Sample Input 2:
40

Sample Output 2:
no

Sample Input 3:
-2345

Sample Output 3:
Invalid Input

```
#include<stdio.h>
#include<stdlib.h>
int checkPerfectSquare(int);
int main(){
	int number=0,res=0;
	scanf("%d",&number);
	res = checkPerfectSquare(number);
	if(res==1)
		printf("yes");
	else if(res==0)
		printf("no");
	else
		printf("invalid Input");
	getchar();
	getchar();
	return 0;
```

```
}


int checkPerfectSquare(int n){
        if(n<0)
                return -1;
        else{
        while (((n % 2) == 0) && n > 1)
                n /= 2;
        return (n == 1);
        }
}
```

47.Find Index

Write a program to find the index of a particular number in a given input array.

Include a function named **findIndex** 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 the corresponding index if the search element is present in the array and returns -1 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 an integer that corresponds to the index of the search element if it is present.
Else, print 'not found'.

Refer sample output for formatting specifications.

Assume that the maximum number of elements in the array is 20 and that all elements in the array are unique.

Sample Input 1:
8

2
1
3
8
6
12
10
19
8

Sample Output 1:
3

Sample Input 2:
8
2
1
3
8
6
12
10
19
80

Sample Output 2:
not found

Sample Input 3:
-5

Sample Output 3:
Invalid Input

Sample Input 4:
5
23
2
-200

Sample Output 4:
Invalid Input

```
#include<stdio.h>

#include<stdlib.h>
```

```
int cricketer[20];
int findIndex(int,int[],int);
int main(){
	int n=0,index=0,input[30],i,search_element=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){
			printf("Invalid Input");
			getchar();
			getchar();
			exit(0);
	}
	}
		scanf("%d",&search_element);
		index = findIndex(n,input,search_element);
			if(index==-1)
				printf("not found");
			else
				printf("%d",index);
		getchar();
```

```
		getchar();
		return 0;
}
	int findIndex( int size,int array[], int search){
	int index,i,f=0;
	for(i=1 ; i<size ; i=i+2)
	{
		if(array[i]==search){
			f=1;
			index = array[i-1];
			break;
		}
	}
	if(f==1)
		return index;
	else
		return -1;
}
```

48.Descending Order Sort

Write a program to sort the given array in descending order.

Include a function named **sortArray** 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 of the elements in the array are 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 array.
Refer sample output for formatting specifications.

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

Sample Input 1:
8
1
6
3
5
8
10
4
9

Sample Output 1:
10
9
8
6
5
4
3
1

Sample Input 2:
-5

Sample Output 2:
Invalid Input

```
#include<stdio.h>

#include<stdlib.h>


void sortArray(int numbers[], int size);

int main(){

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

        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){
                        printf("Invalid Input");
                        getchar();
                        getchar();
                        exit(0);
                }
        }

        sortArray(input,n);
        return 0;
}
void sortArray(int numbers[], int size){
        int i,j,temp=0;
        for(i=0 ; i<size ; i++){
                for(j=i+1 ; j<size ; j++){
                        if(numbers[i]<numbers[j]){
                                temp = numbers[i];
                                numbers[i] = numbers[j];
                                numbers[j] = temp;
                        }
                }
```

```
        }
        for(i=0 ; i<size ; i++)
                printf("%d\n",numbers[i]);
        getchar();
        getchar();

}
```

49.endWithThree

Read the question carefully and follow the input and output format.

Given an input array, Find out the count of numbers that ends with 3.

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 count of numbers that ends with 3.

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 endWithThree(int numbers[], int size) whose return type is integer.

Sample Input 1:
5
23
353
33
12
14
Sample Output 1:
3

Sample Input 2:
5
1
7
3
-8

Sample Output 2:
Invalid input

```
#include<stdio.h>
#include<stdlib.h>

int endWithThree(int numbers[], int size);
int main(){
	int n=0,input[20],i,count=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);
		}
	}

	count = endWithThree(input,n);
	printf("%d",count);
```

```
	getchar();

	getchar();

	return 0;

}

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

	int count=0,i,temp,rem=0;

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

		temp = numbers[i];

		rem = temp%10;

		if(rem==3)

			count++;

	}

	return count;



}
```

50.feeCalculation

Read the question carefully and follow the input and output format.

Student Fees is calculated according to the student's 10th marks. The student will get discount in fees as follows :

Marks discount(%)
>90 - 50%
81-90 - 25%
70-80 - 10%
<70 - 0%
Calculate the fees according to above table.

Note:
Formula : fees - (fees* discount(%))

Include a function named feeCalculation(int fee,int marks) that returns an integer that corresponds to the fee to be paid.

Print "Invalid mark" if the mark is greater than 100
Print "Invalid fee" if the fee is greater than 32767

Print "Invalid input" if any of the input is negative

Input and Output Format :
First line of input represents the fee, second line of input represents the marks of student.

Sample Input 1:
10000
95
Sample Output 1:
5000

Sample Input 2:
15896
101
Sample Output 2:
Invalid mark

```
#include<stdio.h>

#include<stdlib.h>


int feeCalculation(int fee,int marks);

int main(){

	int fees,mark,final_fee=0;

	scanf("%d",&fees);

	scanf("%d",&mark);

	if(mark>100)

		printf("Invalid mark");

	else if(fees>32767)

		printf("Invalid fees");

	else if(mark<0 || fees<0)

		printf("Invalid input");

	else{

	final_fee = feeCalculation(fees,mark);

	printf("%d",final_fee);
```

```
        }
        getchar();
        getchar();
        return 0;
}
int feeCalculation(int fee,int marks){
        int fee_final=0;
        float discount=0.0f;
        if(marks>90)
                discount = .50;
        else if(marks>80 && marks<=90)
                discount = .25;
        else if(marks>=70 && marks<=80)
                discount = .10;
        else
                discount = 0;
        fee_final = fee-(fee*discount);
        return fee_final;
}
```

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:

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;
}
```

61.Duplicate Elements

Tim is working as a data entry staff in a college. His manager wants him to delete the duplicate student id from the entry. Help Tim in writing a program to delete the duplicate elements.

Include a function named **eliminateDuplicate** 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. The output array is stored in a global variable named output1 and the number of elements in the output array is stored in the global variable named output 2.

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 array.
Refer sample output for formatting specifications.

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

Sample Input 1:
8
1
6
3
5
6
8
5
9

Sample Output 1:
1
6
3
5
8
9

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>
void eliminateDuplicate(int[],int);
int main(){
	int n=0,input[20]={0},i;
	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){
		printf("Invalid input");
		getchar();
		getchar();
```

```
                exit(0);
        }
        }

        eliminateDuplicate(input,n);



        return 0;
}
void eliminateDuplicate(int in[],int size){
        int i,j,k=0;
        for(i=0;i<size;i++){
                for(j=i+1;j<size;){
                        if(in[i]==in[j]){
                                for(k=j;k<size;k++)
                                        in[k]=in[k+1];
                                size--;
                        }
                        else
                                j++;
                }
        }
        for(i=0;i<size;i++)
                printf("%d\n",in[i]);
        getchar();
        getchar();
}
```

62.Second Smallest

Write a program to find the second smallest of all divisors of the given number.

For example, the divisors of 21 are 1,3,7 and 21. The second smallest divisor is 3.

Include a function named **secondSmallest** that accepts an integer argument and returns an integer. The function returns the second smallest divisor or returns -1 if it is a negative number or if it is greater than 32767.

If the function returns -1, print Invalid Input.

Input and Output Format:
Input consists of a single integer.
Output consists of a single integer.
Refer sample output for formatting specifications.

Sample Input 1:
21

Sample Output 1:
3

Sample Input 2:
-241

Sample Output 2:
Invalid Input

Sample Input 3:
50000

Sample Output 3:
Invalid Input

```
#include<stdio.h>

#include<stdlib.h>
```

```
int secondSmallest(int);
int main(){
        int number=0,result;
        scanf("%d",&number);
        result = secondSmallest(number);
        if(result!=-1)
                printf("%d",result);
        else
                printf("Invalid input");
        getchar();
        getchar();
        return 0;
}

int secondSmallest(int n){
        int i;
        for(i=2;i<=n;i++){
      if(n%i==0){
                        break;
      }
   }
        return i;
}
```

63.secondMaxMinDiff

Read the question carefully and follow the input and output format.

Given an input array, find the difference b/w second largest and second smallest element in the array.

Hint : There is no repetition of element in the array.

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 difference b/w second largest and second smallest..

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 secondMaxMinDiff(int[] array, int n) whose return type is an integer, which is the difference b/w second largest and second smallest.

Sample Input 1:
5
1
2
3
4
5

Sample Output 1:
2

Sample Input 2:
4
-3

Sample Output 2:
Invalid input

```
#include<stdio.h>

#include<stdlib.h>

int secondMaxMinDiff(int[], int n);

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 = secondMaxMinDiff(input1,size);
        printf("%d",diff);
        getchar();
        getchar();
        return 0;
}
int secondMaxMinDiff(int array[], int n){
        int i=0,j=0,diff=0,temp=0;
        for(i=0;i<n;i++){
            for(j=i+1;j<n;j++){
                if(array[i]<array[j]){
                    temp = array[i];
                    array[i]=array[j];
                    array[j]=temp;
                }
            }
        }
        diff=array[1]-array[n-2];
        return diff;
}
```

64.newNumber

Read the question carefully and follow the input and output format.

Write a program to find the difference between consecutive digits in the given input integer and display it.

Input and Output Format:
Input consists of an integer and output the difference between the consecutive digits.

Print "Number too small" if the number is less than 0
Print "Number too large" if the number is greater than 32767

Include a function named newNumber(int number) that returns a integer

Sample Input 1:
1325
Sample Output 1:
213

Sample Input 2:
-13
Sample Output 2:
Number too small

```
#include<stdio.h>

#include<stdlib.h>

int newNumber(int number);

int main(){

        int num=0,new_num=0;

        scanf("%d",&num);

        if(num>32767)

                printf("Number too large");

        else if(num<0)

                printf("Number too small");

        else{

                new_num=newNumber(num);

                printf("%d",new_num);

        }

        getchar();

        getchar();
```

```
	return 0;
}
int newNumber(int number){
	int rem,i=1,rem1=0,diff=0,digit=0;
	while(number>10){
		rem=number%10;
		number=number/10;
		rem1=number%10;
		if(rem>rem1)
			diff=rem-rem1;
		else
			diff=rem1-rem;
		digit=digit+(diff*i);
		i=i*10;
	}
	return digit;
}
```

65.Digit Counting

In a game show everybody got one coupon with some code. They need to count the digits in the code and send SMS to the given number.

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

Include a function named **countDigits** that accepts an integer argument and returns an integer that corresponds to the number of digits. If the input is a negative number, the function returns -1.

If the function returns -1, print Invalid Input.

Input and Output Format:
Input consists of a single integer.

Output consists of an integer that corresponds to the number of digits in the input.

Sample Input 1:
250

Sample Output 1:
3

Sample Input 2:
-2345

Sample Output 2:
Invalid Input

```
#include<stdio.h>
#include<stdlib.h>
int countDigits(int number);
int main(){
	int num=0,digits=0;
	scanf("%d",&num);
	digits = countDigits(num);
	if(digits==-1)
		printf("invalid input");
	else
		printf("%d",digits);
	getchar();
	getchar();
	return 0;
}
int countDigits(int number){
	int rem,count=0;
```

```
        if(number<0)
                count=-1;
        else
        {
        while(number!=0){
                count++;
                number=number/10;
        }
        }
        return count;
}
```

66.maxScoreCount

Read the question carefully and follow the input and output format.

Student1 and Student2 are of same class and have recieved their scores for different subjects. Given each subject scores of the Student1 & Student2, Find out in how many subjects student1 has scored more marks than student2.

Input and Output Format :
First line of input corresponds to the n, the number of subjects. The next n lines correspond to the scores of Student 1 and the next n lines correspond to the scores of student 2. Output is the number of subjects student1 scored more marks than student2.

Print "Invalid size" when size of the array is a negative number and terminate the program
Print "Invalid score" when there is any negative score and terminate the program

Include a function named maxScoreCount(int size,int student1[],int student2[]) that returns an integer, the number of subjects student1 scored more marks than student2

Sample Input 1:
7
45
23
67
34
88
13
67

33
56
89
44
67
89
55

Sample Output 1:
3

Sample Input 2:
-3
Sample Output 2:
Invalid size

Sample Input 3:
2
30
-60
Sample Output 3:
Invalid score

```
#include<stdio.h>

#include<stdlib.h>

int maxScoreCount(int size,int student1[],int student2[]);

int main(){

	int size=0,input1[20],input2[20],i=0,count=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);
            }
        }
        count = maxScoreCount(size,input1,input2);
        printf("%d",count);
        getchar();
        getchar();
        return 0;
}
int maxScoreCount(int size,int student1[],int student2[]){
        int i=0,count=0;
        for(i=0;i<size;i++){
            if(student1[i]>student2[i])
                count++;
        }
        return count;
}
```

67.nextPrime

Read the question carefully and follow the input and output format.

Write a program to find out the Next Prime to the given number.

Hint: number is always less than 100.

Input and Output Format :

First line of input consists of n, the number. Output is a single integer that displays the next prime.

Print "Number too large" when the given input number is greater than 32767 .
Print "Number too small" when given input is a negative number.

Include a function named nextPrime(int num) whose return type is an integer, the next prime.

Sample Input 1:
9

Sample Output 1:
11

Sample Input 2:
98987

Sample Output 2:
Number too large

```
#include<stdio.h>

#include<stdlib.h>

int nextPrime(int num);

int main(){

        int number=0,result;

        scanf("%d",&number);

        if(number<0)

                printf("number too small");

        else if(number>32767)

                printf("number too large");

        else{

        result = nextPrime(number);

        printf("%d",result);
```

```
	}
	getchar();
	getchar();
	return 0;
}

int nextPrime(int num){
	int next=0,i,m,c=0;
	for(i=num+1;i<100;i++){
		c=0;
		for(m=1;m<=i;m++){
			if(i%m==0)
				c++;
		}
		if(c==2)
			next=i;
		if(next!=0)
			break;
	}
	return next;
}
```

68.Array Average

Write a program to find the average of the array.

Include a function named **avgArray** that accepts 2 arguments and returns a float. 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 a float that corresponds to the average 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 a floating point number that corresponds to the average. It is displayed correct to 2 decimal places.

Assume that the maximum size of the array is 20.

Sample Input 1:

7
1
9
8
4
6
4
5

Sample Output 1:

5.29

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>

float avgArray(int numbers[], int size);

int main(){
```

```
	int size=0,input1[20],i=0;
	float avg=0.0f;
	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);
		}
	}
	avg = avgArray(input1,size);
	printf("%.2f",avg);
	getchar();
	getchar();
	return 0;
}
float avgArray(int numbers[], int size){
	int i=0,j=0,max=0,sum=0;
	float avg=0.0f;
```

```
        for(i=0;i<size;i++){
                sum=sum+numbers[i];
        }
        avg=(float)sum/size;
        return avg;
}
```

69.Strong Number

Write a program to find whether the given input number is a Strong Number

Strong Number : (In a number sum of Factorial of individual digits equals to the same number).

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

1. 1 if the input is a Strong Number
2. 0 if the input is not a Strong Number
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:
145

Sample Output 1:
yes

Sample Input 2:
141

Sample Output 2:
no

Sample Input 3:
-2345

Sample Output 3:

Invalid Input

```
/*
num1=num;
 while(num){
  i=1,f=1;
  r=num%10;

  while(i<=r){
   f=f*i;
   i++;
  }
  sum=sum+f;
  num=num/10;

 }
 if(sum==num1)
  printf("%d is a strong number",num1);
 else
  printf("%d is not a strong number",num1);

*/

#include<stdio.h>
#include<stdlib.h>
int checkStrong(int);
int main(){
```

```
	int number=0,result;
	scanf("%d",&number);
	result = checkStrong(number);
	if(result==1)
		printf("yes");
	else if(result==0)
		printf("no");
	else
		printf("Invalid input");
	getchar();
	getchar();
	return 0;
}

int checkStrong(int num){
	int res=0,f=0,r=0,num1=0,i,sum=0;
	if(num<0)
		res=-1;
	else{
		num1=num;
		while(num){
			i=1,f=1;
			r=num%10;
			while(i<=r){
				f=f*i;
				i++;
			}
```

```
                sum=sum+f;
                num=num/10;
        }
        if(sum==num1)
                res=1;
        else
                res=0;
    }
    return res;
}
```

70.Sum of Even Digits

In the computer science department, HOD wants to divide students into two groups based on their roll numbers. He decided to find the sum of the even digits in the roll number of each student and decided to split them into 2 groups based on this. Write a program to find the sum of the even digits in a number.

Include a function named **addEvenDigits** to find the sum of even digits in a number. This function accepts an integer argument and returns an integer. The function returns -1 if the roll number is less than zero or if the roll number is greater than 32767. Refer function specifications given at the end of the problem for further details.

If the roll number is less than 0 or if it exceeds 32767, print “Invalid Input”.

Input and Output Format:
The input consists of an integer that corresponds to the roll number.
The output consists of an integer that corresponds to the sum of the even digits in the roll number.

Sample Input 1:
3487

Sample Ouput 1:
12

Sample Input 2:
-8

Sample Output 2:

Invalid Input

```
#include<stdio.h>
#include<stdlib.h>
int addEvenDigits(int);
int main(){
	int number,result=0;
	scanf("%d",&number);
	result = addEvenDigits(number);
	if(result==-1)
		printf("Invalid input");
	else
		printf("%d",result);
	getchar();
	getchar();
	return 0;
}

int addEvenDigits(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;
```

```
                    n=n/10;
                }
            }
            return res;
}
```

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();

}
```

81.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;
}
```

82.clearedStage1

Read the question carefully and follow the input and output format.

Given an integer array. The first index represents the Student id, Second index represents C-programming marks and the third index Represents SQL marks. Write a program to find the Ids of students who have cleared both C-programming and SQL.

Note :(1) The Pass Marks is >=70

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 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 number available in the input array and terminate the program.

Include a function named clearedStage1(int array[], int size) whose return type is void.
The output array is stored in a global variable named cleared.

Sample Input 1:
9
1
25
75
3
75
80
2
75
75

Sample Output 1:
3
2

Sample Input 2:
6
4
25
-78

Sample Output 2:
Invalid input

```
#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;
}
```

83.findLargest

Read the question carefully and follow the input and output format.

Write a program to find the largest of the 3 given numbers.

Input and Output Format :

Input consists of 3 integers. Output consist of an integer that is the maximum.

Print "Number too large" when any of given input numbers is greater than 32767 .
Print "Number too small" when given input is a negative number.

Include a function named findLargest(int num1, int num2, int num3) whose return type is an integer, which is the largest.

Sample Input 1:
2
3
4

Sample Output 1:
4

Sample Input 2:
98974

Sample Output 2:
Number too large

Sample Input 3:
-32767

Sample Output 3:
Number too small

84.sumThreeLargest

Read the question carefully and follow the input and output format.

Write a program to find the sum of first ,second and third largest element in the given array.

Input and Output Format:
First line of input consists of n, the number of elements. Next n lines correspond to the array elements .
Output consists of an Integer, 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 sumThreeLargest(int array[], int n) whose return type is integer

Sample Input 1:
8
1
2
2
3
4
5
5

4

Sample Output 1:
12

Sample Input 2:
4
1
2
-3
Sample Output 2:
Invalid input

```
#include<stdio.h>

#include<stdlib.h>

int sumThreeLargest(int array[], int n);

int main(){

	int n=0,input[20]={0},i,largest;

	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);

	}
```

```
        }


        largest = sumThreeLargest(input,n);
        printf("%d",largest);
        getchar();
        getchar();
        return 0;
}
int sumThreeLargest(int array[], int n){
        int i,j,k=0,temp=0;
        for(i=0;i<n;i++){
                for(j=i+1;j<n;){
                        if(array[i]==array[j]){
                                for(k=j;k<n;k++)
                                        array[k]=array[k+1];
                                n--;
                        }
                        else
                                j++;
                }
        }
        for(i=0;i<n;i++){
                for(j=i+1;j<n;j++){
                        temp = array[i];
                        array[i] = array[j];
                        array[j] = temp;
                }
```

```
    }

    return array[0]+array[1]+array[2];

}
```

85.highestProfitYear

Read the question carefully and follow the input and output format.

An array holds information of a company profit margin and year. Find out the year in which highest revenue was earned. Assume the first index of array indicates year and the next index indicates the amount of money earned by the company and so on.

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 highestProfitYear(int revenue[],int size) whose return type is an integer, which is the year.

Sample Input 1:
6
2012
10000
2011
5000
2009
4000

Sample Output 1:
2012

Sample Input 2:
8
2015
89745
-2015

Sample Output 2:
Invalid input

```
#include<stdio.h>

#include<stdlib.h>

int highestProfitYear(int revenue[],int size);
```

```
int main(){
	int n=0,input[20]={0},i,highest;
	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);
	}
	}

	highest = highestProfitYear(input,n);
	printf("%d",highest);
	getchar();
	getchar();
	return 0;
}
int highestProfitYear(int revenue[],int size){
	int i,j,k=0,highest=0,max=0;
	for(i=1;i<size;i=i+2){
		if(revenue[i]>max){
```

```
                    max=revenue[i];
                    highest=revenue[i-1];
                }
        }

        return highest;
}
```

86.countNoOfConnections

Read the question carefully and follow the input and output format.

Given an input array, the elements are of format XYYY . where X represents the connection type. YYY represents the connection id.

If X is 2 -> means 2G connection
If X is 3 -> means 3G connection
If X is 4 -> means 4G connection
You need to find the number of type of connections.

Note:
If a particular connection type (starting with 2 or 3 or 4) is not available represent with zero in the corresponding position.

Include a function named countNoOfConnections(int connection_list[],int no) that returns the number of types of connections

Business Rules :
1) Print "Invalid array size" when size of the array is a negative number and terminate the program.
2) Print "Invalid connection" when there is any negative number available in the input array and terminate the program

Input and Output Format :
First line of input corresponds to n, next n lines corresponds to the elements of the array
Output consists of the the number of type of connections. [1st line of the output corresponds to the number of 2G connections, 2nd line corresponds to the number of 3G connections and 3rd line corresponds to the number of 4G connections]

Sample Input 1 :
5
2333
3101
2102
4567
3123

Sample Output 1:
2
2
1

Sample Input 2 :
2
-2234

Sample Output 2:
Invalid connection

```
#include<stdio.h>

#include<stdlib.h>

void countNoOfConnections(int connection_list[],int no);

int main(){

	int n=0,input[20]={0},i;

	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 Connection");

		getchar();

		getchar();

		exit(0);

	}
```

```
        }

        countNoOfConnections(input,n);

        getchar();
        getchar();
        return 0;
}
void countNoOfConnections(int connection_list[],int no){
        int i,j,k=0,temp=0,x=0,two=0,three=0,four=0,other=1,out[10]={0};
        for(i=0;i<no;i++){
                temp=connection_list[i];
                x=temp/1000;

                if(x==2)
                        two++;
                else if(x==3)
                        three++;
                else if(x==4)
                        four++;
                else
                        other=0;
        }
        out[0]=two;
        out[1]=three;
        out[2]=four;
```

```
    for(i=0;i<3;i++)
        printf("%d\n",out[i]);
}
```

87.Sum of positive numbers in Array

Write a program to find the sum of positive numbers in an array.

Include a function named **addPositives** 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 the sum of the positive numbers in the array.

If the size of 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.
Assume that the maximum size of the array is 20.

Sample Input 1:
5
3
5
-2
6
-6

Sample Output 1:
14

Sample Input 2:
-5

Sample Output 2:
Invalid Input

```
#include<stdio.h>

#include<stdlib.h>

int addPositives(int array[],int n);
```

```
int main(){
	int n=0,input[20]={0},i,sum;
	scanf("%d",&n);
	if(n<0){
		printf("Invalid input");
		getchar();
		getchar();
		exit(0);
	}
	for(i=0;i<n;i++)
		scanf("%d",&input[i]);




	sum=addPositives(input,n);
	printf("%d",sum);
	getchar();
	getchar();
	return 0;
}
int addPositives(int array[],int n){
	int i,j,sum=0;
	for(i=0;i<n;i++){
		if(array[i]>0)
			sum=sum+array[i];
	}
	return sum;
```

}

88.Find Index

Write a program to find the index of a particular number in a given input array.

Include a function named **findIndex** 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 the corresponding index if the search element is present in the array and returns -1 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 an integer that corresponds to the index of the search element if it is present.
Else, print 'not found'.

Refer sample output for formatting specifications.

Assume that the maximum number of elements in the array is 20 and that all elements in the array are unique.

Sample Input 1:
8
2
1
3
8
6
12
10
19
8

Sample Output 1:

3

Sample Input 2:
8
2
1
3
8
6
12
10
19
80

Sample Output 2:
not found

Sample Input 3:
-5

Sample Output 3:
Invalid Input

Sample Input 4:
5
23
2
-200

Sample Output 4:
Invalid Input

```
#include<stdio.h>
#include<stdlib.h>
int cricketer[20];
int findIndex(int,int[],int);
int main(){
        int n=0,index=0,input[30],i,search_element=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){
			printf("Invalid Input");
			getchar();
			getchar();
			exit(0);
	}
	}
		scanf("%d",&search_element);
		index = findIndex(n,input,search_element);
			if(index==-1)
				printf("not found");
			else
				printf("%d",index);
		getchar();
		getchar();
		return 0;
}
	int findIndex( int size,int array[], int search){
```

```
    int index,i,f=0;
    for(i=1 ; i<size ; i=i+2)
    {
        if(array[i]==search){
            f=1;
            index = array[i-1];
            break;
        }
    }
    if(f==1)
        return index;
    else
        return -1;
}
```

89.Store Consequtives

Write a program to obtain a new array that contains the consequtive values of the given input array. The output array is named as output1.

Include a function named **storeConsequtives** 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 . The output array is stored in a global variable named output1.

If the size of the array is negative or if any of the elements in the array are 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 array.
Refer sample output for formatting specifications.

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

Sample Input 1:

4
2
5
1
4

Sample Output 1:

3
6
2
5

Sample Input 2:

-5

Sample Output 2:

Invalid Input

```
#include<stdio.h>
#include<stdlib.h>
int output1[20];
void storeConsequtives(int numbers[], 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);
		}
	}
	storeConsequtives(input1,size);
	return 0;
}
void storeConsequtives(int numbers[], int size){
	int i=0,j=0;
	for(i=0;i<size;i++){
		output1[i]=numbers[i]+1;
		printf("%d\n",output1[i]);
	}
	getchar();
	getchar();
}
```

90.findGrade

Read the question carefully and follow the input and output format.

In a school examination the result of students are published in the form of an array where first index is the student id and the second index is the total marks in mathematics third index is student id and fourth index is total marks in mathematics and so on... Write a method that assigns the student id as the key and the grade in mathematics as the value to the output array based on the following conditions:

If(Marks >=90) : 1
If(Marks >=80 and <90) : 2
If(Marks >=70 and <80) : 3
If(Marks <70) : 0

Hint: Array size is always is even.

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 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 findGrade(int array[], int size) whose return type is void.
The output array is stored in a global variable named grade.

Sample Input 1:
10
1
65
2
74
3
86
4
95
6
69
Sample Output 1:
1
0
2
3
3
2
4
1
6
0

Sample Input 2:
4
-3
Sample Output 2:
Invalid input

```
#include<stdio.h>

#include<stdlib.h>

int grade[20];

void findGrade(int array[], 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);
		}
	}
	findGrade(input1,size);
	return 0;
}
void findGrade(int array[], int size){
	int i=0,j=0;
	for(i=1;i<size;i=i+2){
		if(array[i]>=90){
			grade[i-1]=array[i-1];
			grade[i]=1;
```

```
            }
            else if(array[i]>=80 && array[i]<90){
                    grade[i-1]=array[i-1];
                    grade[i]=2;
            }
            else if(array[i]>=70 && array[i]<80){
                    grade[i-1]=array[i-1];
                    grade[i]=3;
            }
            else{
                    grade[i-1]=array[i-1];
                    grade[i]=0;
            }
    }
    for(i=0;i<size;i++)
            printf("%d\n",grade[i]);
    getchar();
    getchar();
}
```

91.Odd Even Average

The Owner of a block visited the Layout and found that he has some plot numbers of his own and some are odd numbers and some are even numbers. He is maintaining the details in a file in the system. For the password protection our owner has followed one formula. He calculated the sum of his even numbers plot and sum of odd numbers plot and found the average of those two and he used that average as his password for the details file. Find the password that our owner has arrived.

Include a function named **avgOddEvenSum** that accepts 2 arguments and returns a float. 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 a float that corresponds to the average 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 a floating point number that corresponds to the average. It is displayed correct to 2 decimal places.
Assume that the maximum size of the array is 20.

Sample Input 1:
5
1
2
3
4
5

Sample Output 1:
7.50

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>

float avgOddEvenSum(int[],int);
```

```
int main(){
	int n=0,i,flag=0,input[20];
	float avg=0.0f;
	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);
	}
	avg = avgOddEvenSum(input,n);
	printf("%0.2f",avg);
	getchar();
	getchar();
	return 0;
}
```

```
float avgOddEvenSum(int array[],int size){
	float avg;
	int i,sumodd=0,sumeven=0;
	for(i=0;i<size;i++){
		if(array[i]%2==0)
			sumeven=sumeven+array[i];
		else
			sumodd=sumodd+array[i];
	}
	avg = float(sumeven+sumodd)/2.0f;
	return avg;
}
```

92.convertToBinary

Read the question carefully and follow the input and output format.

Kate is a military officer. He needs to send top-secret code to the other soldiers in a different place. He need to encrypt it. We need to write a function to convert the given decimal number to the corresponding binary number.

Input consist of single integer. Output consists of binary number.

1) Print "Number too small" when input is negative.
2) Print "Number too large" when input value is greater than 100.

Include a function named convertToBinary(int num) whose return type is void.

Sample Input 1:
12

Sample Output 1:
1100

Sample Input 2:
101

Sample Output 2:
Number too large

```
#include<stdio.h>
#include<stdlib.h>
void convertToBinary(int num);
int main(){
        int number=0;
        scanf("%d",&number);
        if(number<0)
                printf("Number too small");
        else if(number>100)
                printf("Number too large");
        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();
}
```

93.reverseNumber

Read the question carefully and follow the input and output format.

Write a program to find the reverse of a given input integer

Input and Output Format :
Input consists of an integer, n. Output consist of the reverse of the number n.

Print "Number too large" when the given input number is greater than 32767
Print "Number too small" when the given input numbers is a negative number.

Include a function named reverseNumber(int num) whose return type is integer, the reverse of n.

Sample Input 1:
1234

Sample Output 1:
4321

Sample Input 2:
326357

Sample Output 2:
Number too large

```
#include<stdio.h>

#include<stdlib.h>

int reverseNumber(int num);

int main(){

        int number=0,revnumber=0;

        scanf("%d",&number);

        if(number<0)

                printf("Number too small");

        else if(number>32767)

                printf("Number too large");

        else{

                revnumber = reverseNumber(number);

                printf("%d",revnumber);

        }

        getchar();
```

```
	getchar();
	return 0;
}
int reverseNumber(int num){
	int reverse=0;
	    while (num != 0)
		{
			reverse = reverse * 10;
			reverse = reverse + num%10;
			num=num/10;
		}
		 return reverse;
}
```

94.Sum of Prime Cubes

Given an input integer n, write a program to find the sum of the cubes of the prime numbers upto n. (including n)

Please note that 1 is neither prime nor composite.

Include a function named **sumCubeOfPrime** that accepts an integer argument and returns an integer. The function returns -1 if the input is a negative number or if it is greater than 3000.

If the function returns -1, print Invalid Input.

Input and Output Format:
Input consists of a single integer.
Output consists of a single integer.
Refer sample output for formatting specifications.

Sample Input 1:
5

Sample Output 1:
160

Sample Input 2:
-241

Sample Output 2:
Invalid Input

Sample Input 3:
50000

Sample Output 3:
Invalid Input

```
#include<stdio.h>
#include<stdlib.h>
int sumCubeOfPrime(int);
int main(){
	int number,result=0;
	scanf("%d",&number);
	result = sumCubeOfPrime(number);
	if(result==-1)
		printf("Invalid input");
	else
		printf("%d",result);
	getchar();
	getchar();
	return 0;
}

int sumCubeOfPrime(int n){
	int i,count=0,num,sum=0;
	if(n<0 || n>3000)
```

```
            sum=-1;
        else{

        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*num);

        }
        }
        return sum;
}
```

95.Negative Elements

Write a program to remove all the negative elements of the input array , sort the positive elements and stores them in an output array .

Include a function named **eliminateNegatives** 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. The output array is stored in a global variable named output1 and the size of the output array is stored in a global variable named output2.

If the size of 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 array.
Refer sample output for formatting specifications.

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

Sample Input 1:

8
1
6
3
5
-6
8
-15
9

Sample Output 1:

1
3
5

6
8
9

Sample Input 2:

-5

Sample Output 2:

Invalid Input

```
 #include<stdio.h>
#include<stdlib.h>
int output1[20];
void eliminateNegatives(int[],int);
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]);

        eliminateNegatives(input1,size);

        return 0;
}
void eliminateNegatives(int array[],int size){
        int i=0,k=0,j=0,temp=0;
        for(i=0;i<size;i++){
                if(array[i]>0){
                        output1[k]=array[i];
                        k++;
                }
        }
        for(i=0;i<k;i++){
                for(j=i+1;j<k;j++){
                        if(output1[i]>output1[j]){
                        temp = output1[i];
                        output1[i]=output1[j];
                        output1[j]=temp;
                        }
                }
        }
        for(i=0;i<k;i++)
                printf("%d\n",output1[i]);
        getchar();
        getchar();
}
```

96.Savings Calculation

Jim got salary for this month and he spends 50% of his salary for food and 20% of his salary for travel. If the number of days he worked is 31 he gets a bonus of Rs.500. Write a program to find how much he can save in his pocket after spending all these?

Include a function named **calculateSavings** that accepts 2 integer arguments and returns a float. The first integer corresponds to Jim's basic salary and the second integer corresponds to the number of days Jim has worked. The function returns a float that corresponds to the amount that Jim could save.

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

1. Basic salary is greater than 9000
2. Number of working days is greater than 31
3. Basic salary is negative
4. Number of working days is negative

Input and Output Format:

Input consists of 2 integers. The first integer corresponds to Jim's basic salary and the second integer corresponds to the number of days he has worked.

Output consists of a single float that corresponds to Jim's savings. Jim's savings is displayed correct to 2 decimal places.

Sample Input 1:

7000

30

Sample Output 1:

2100 .00

Sample Input 2:

50000

Sample Output 2:

Invalid Input

```
#include<stdio.h>

#include<stdlib.h>

float calculateSavings(int,int);

int main(){

        int basic=0,days=0;

        float save=0.0f;

        scanf("%d%d",&basic,&days);

        if(basic>9000 || days>31 || basic<0 || days<0){

                printf("Invalid input");
```

```
                getchar();
                getchar();
                exit(0);
        }
        save = calculateSavings(basic,days);
        printf("%0.2f",save);
        getchar();
        getchar();
        return 0;
}

float calculateSavings(int basic,int days){
        float spent=0.0f,savings=0.0f;
        spent=(float)(basic*70)/100;
        if(days==31)
                savings = basic+500-spent;
        else
                savings = basic-spent;
        return savings;
}
```

97.studentMarks

Read the question carefully and follow the input and output format.

The Given input is of the format XXXYY , where XXX is Id , YY is marks. Write a code to display the id and marks separately as given in the output formats. [Refer sample input and output]

Input and Output Format :
Input consists of a number. Refer sample output for output format.

Print "Number too large" when any of given input numbers is greater than 32767 .
Print "Number too small" when given input is a negative number.

Include a function named studentMarks(int number) whose return type is void.

Sample Input 1:
12345

Sample Output 1:
123
45

Sample Input 2:
-13

Sample Output 2:
Invalid input

```
#include<stdio.h>
#include<stdlib.h>
void studentMarks(int number);
int main(){
	int number=0,result;
	scanf("%d",&number);
	if(number<0)
		printf("number too small");
	else if(number>32767)
		printf("number too large");
	else
		studentMarks(number);
	getchar();
	getchar();
	return 0;
}

void studentMarks(int number){
	int id=0,mark=0,rem=0,i=1,flag=1,j=1;
	while(number!=0){
```

```
            if(flag<=2){
                rem=number%10;
                mark=mark+(rem*i);
                i=i*10;
                number=number/10;
                flag++;
            }
            else{
                rem=number%10;
                id=id+(rem*j);
                j=j*10;
                number=number/10;
                flag++;
            }
        }
        printf("%d\n%d",id,mark);
        getchar();
        getchar();
}
```

98.maximumSum

Read the question carefully and follow the input and output format.

Given an Integer array, find out sum of Even and odd Numbers individually and find the maximum.

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 maximum of odd and even 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 maximumSum(int numbers[], int size) whose return type is an integer,.

Sample Input 1:

5
12
13
14
15
16

Sample Output 1:
42

Sample Input 2:
-13

Sample Output 2:
Invalid array size

```
#include<stdio.h>
#include<stdlib.h>
int maximumSum(int[],int);
int main(){
	int n=0,i,flag=0,input[20],max=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);
        }
        max = maximumSum(input,n);
        printf("%d",max);
        getchar();
        getchar();
        return 0;
}
int maximumSum(int numbers[], int size){
        int i=0,max=0,evensum=0,oddsum=0;
        for(i=0;i<size;i++){
                if(numbers[i]%2==0)
                        evensum = evensum + numbers[i];
                else
                        oddsum = oddsum + numbers[i];
                if(evensum>oddsum)
                        max = evensum;
                else
                        max=oddsum;
        }
        return max;
}
```

99.newNumber

Read the question carefully and follow the input and output format.

Write a program to find the difference between consecutive digits in the given input integer and display it.

Input and Output Format:

Input consists of an integer and output the difference between the consecutive digits.

Print "Number too small" if the number is less than 0
Print "Number too large" if the number is greater than 32767

Include a function named newNumber(int number) that returns a integer

Sample Input 1:
1325
Sample Output 1:
213

Sample Input 2:
-13
Sample Output 2:
Number too small

```
#include<stdio.h>

#include<stdlib.h>

int newNumber(int number);

int main(){

	int num=0,new_num=0;

	scanf("%d",&num);

	if(num>32767)

		printf("Number too large");

	else if(num<0)

		printf("Number too small");

	else{

		new_num=newNumber(num);

		printf("%d",new_num);

	}

	getchar();

	getchar();

	return 0;

}

int newNumber(int number){
```

```
    int rem,i=1,rem1=0,diff=0,digit=0;
    while(number>10){
        rem=number%10;
        number=number/10;
        rem1=number%10;
        if(rem>rem1)
            diff=rem-rem1;
        else
            diff=rem1-rem;
        digit=digit+(diff*i);
        i=i*10;
    }
    return digit;
}
```

100.nonWorkingDoctors

Read the question carefully and follow the input and output format.

A doctor survey results information is stored in 2 arrays. First array represents all doctors ids (working and non -working both). Second array represents only working doctor's id . Please find the doctor ids who are not working .

Input and Output Format :

First line of input corresponds to n1, the size of first array and next n1 lines correspond to the elements of the first array. The next line corresponds to n2, the size of second array and next n2 lines correspond to the elements of the second array

Output is the id's of doctor who are not working

Print "Invalid array size" when size of the array is a negative number and terminate the program

Print "Invalid id" when there is any negative numbers available in the input array and terminate the program.

Include a function named nonWorkingDoctors(int total[],int working[],int n,int m) whose return type is void.

Sample Input 1:

```
7
7
2
```

3
4
5
6
1
3
3
4
5

Sample Output 1:

7
2
6
1

Sample Input 2:

7
7
2
3
4
5
6
-1

Sample Output 2:

Invalid id

```
#include<stdio.h>

#include<stdlib.h>

int output1[20];

void nonWorkingDoctors(int total[],int working[],int n,int m);

int main(){

        int size1=0,size2=0,input1[20],input2[20],i=0;

        scanf("%d",&size1);

        if(size1<0){

                printf("Invalid array size");

                getchar();

                getchar();

                exit(0);
```

```
}
for(i=0;i<size1;i++){
        scanf("%d",&input1[i]);
        if(input1[i]<0){
                printf("Invalid id");
                getchar();
                getchar();
                exit(0);
        }
}
scanf("%d",&size2);
if(size2<0){
        printf("Invalid array size");
        getchar();
        getchar();
        exit(0);
}
for(i=0;i<size2;i++){
        scanf("%d",&input2[i]);
        if(input2[i]<0){
                printf("Invalid id");
                getchar();
                getchar();
                exit(0);
        }
}
nonWorkingDoctors(input1,input2,size1,size2);
```

```
	return 0;
}
void nonWorkingDoctors(int total[],int working[],int n,int m){
	int i=0,j=0,k=0,flag=0;
	for(i=0;i<n;i++){
		flag=0;
		for(j=0;j<m;j++){
			if(total[i]==working[j])
				flag=1;
		}
		if(flag==0)
			printf("%d\n",total[i]);
	}
	getchar();
	getchar();
}
```

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;
}
```