Luxor University

Faculty of computers and information

Programming Fundamentals

Lab Sheet #5

Objectives:

- Learn how to think on paper before coding.
- Learn how to construct your program step by step in detail (Algorithm).
- Learn how to present your program algorithm in an efficient and organized way.
- Start you first C console programs.
- Learn how to solve problems using (variables & data types, input & output and operators).
- Understand the difference between sequential execution and transfer of controls and loop flow.
- Learn how to solve problems using (conditions).
- Learn how to solve problems using loops (for, while, do-while)

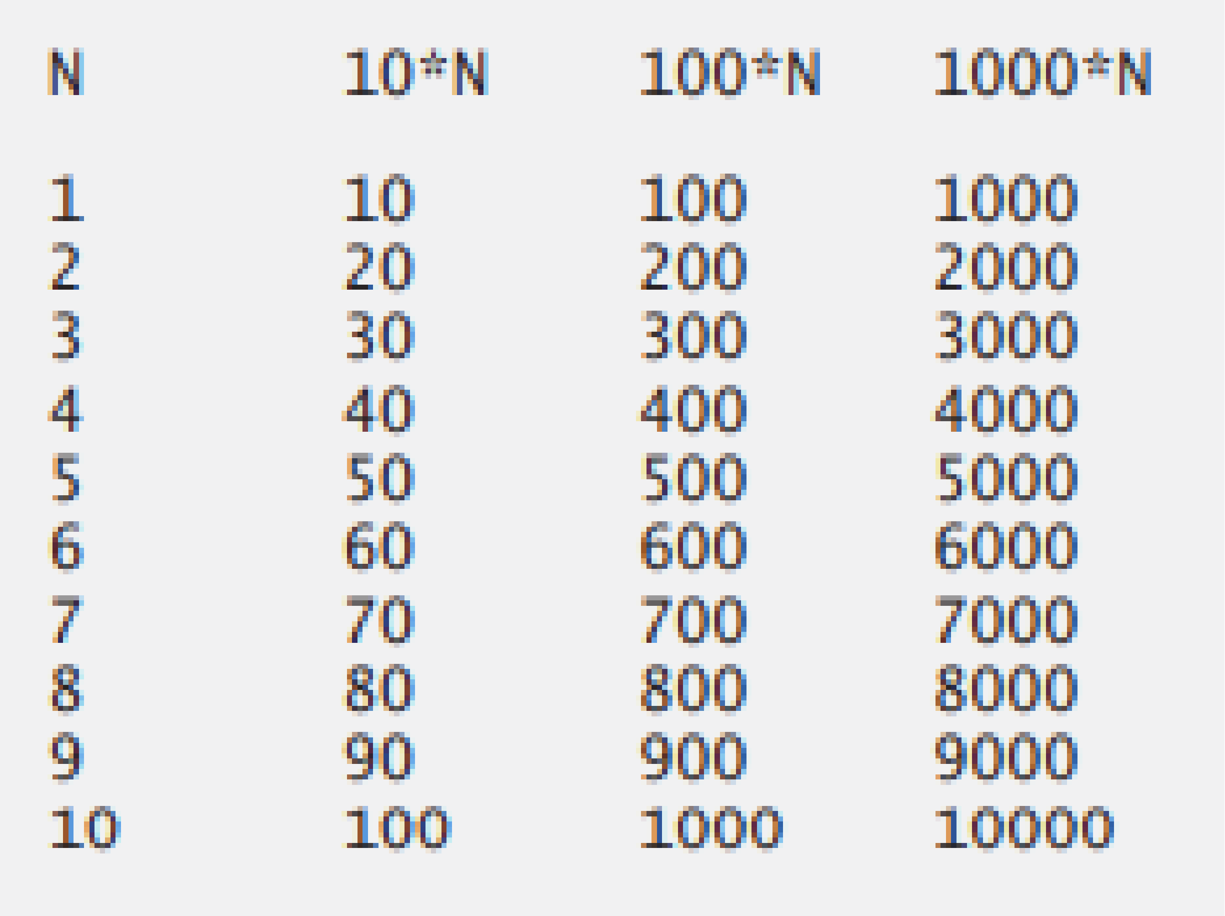
Problems:

1. Tabular Output
2. Printing shapes
3. Sequences
4. Binary Equivalent of a Decimal Number
5. Odd & even
6. Find the Smallest
7. SUM Till >=200
8. Sales Commission Calculator
9. Salary Calculator
10. Calculating Sales
11. Factorial
12. Table of Decimal, Binary, Octal and Hexadecimal Equivalents
13. GCD
14. Super Mario
15. Armstrong Numbers
16. Fibonacci series

Tabular Output

Problem statement:

Write a program that uses looping to print the following table of values. Use the tab escape sequence (\t), in the printf statement to separate the columns with tabs.



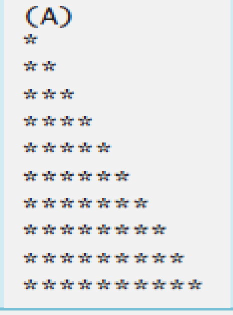
```
N       10*N    100*N   1000*N

1       10      100     1000
2       20      200     2000
3       30      300     3000
4       40      400     4000
5       50      500     5000
6       60      600     6000
7       70      700     7000
8       80      800     8000
9       90      900     9000
10      100     1000    10000
```

Printing shapes

Problem statement:

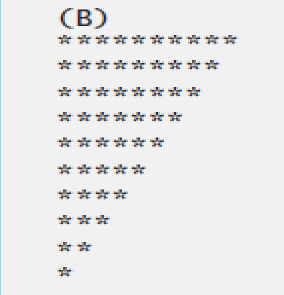
Write a program that prints the following patterns separately. Use **for** loops to generate the patterns

Shape 1



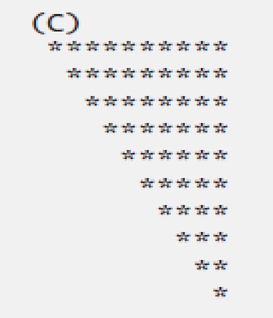
```
(A)
*
**
***
****
*****
******
*******
********
*********
**********
```

Shape 2



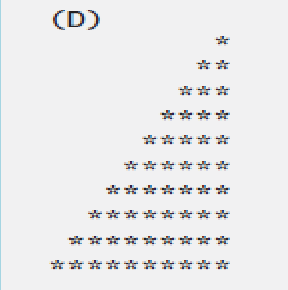
```
(B)
**********
*********
********
*******
******
*****
****
***
**
*
```

Shape 3



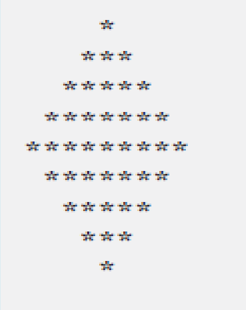
```
(C)
**********
 *********
  ********
   *******
    ******
     *****
      ****
       ***
        **
         *
```

Shape 4



```
(D)
         *
        **
       ***
      ****
     *****
    ******
   *******
  ********
 *********
**********
```

Shape 5



```
    *
   ***
  *****
 *******
*********
 *******
  *****
   ***
    *
```

.

Sequences

Problem statement:

Write **for** statements that print the following sequences of values:

- 1, 2, 3, 4, 5, 6, 7, … ,999,1000
- 3, 8, 13, 18, 23, …, 993,998
- 20, 14, 8, 2, –4, –10, …, -994, -1000
- 3, 15, 75, 375, …, 46875, 234375
- 1, 2, 4, 8, 16, …, 4096, 8192

Binary Equivalent of a Decimal Number

Problem statement:

Write a program to print the binary equivalent of a given decimal integer.

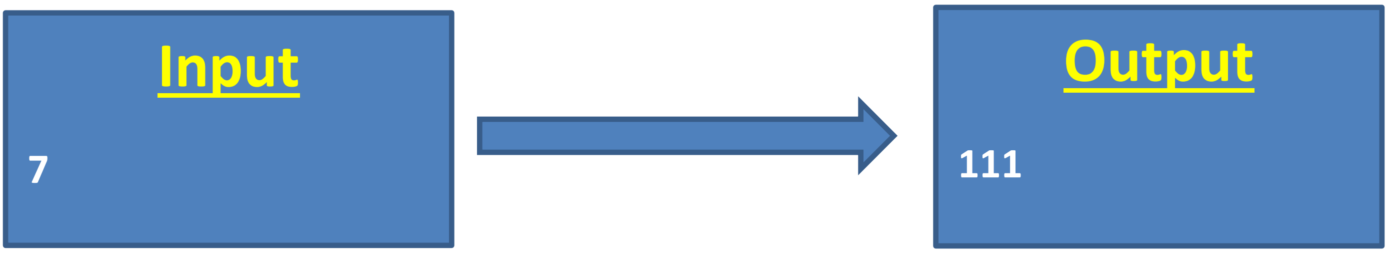
Input:

Decimal integer (x).

Output:

binary equivalent of (x).

Example 1:



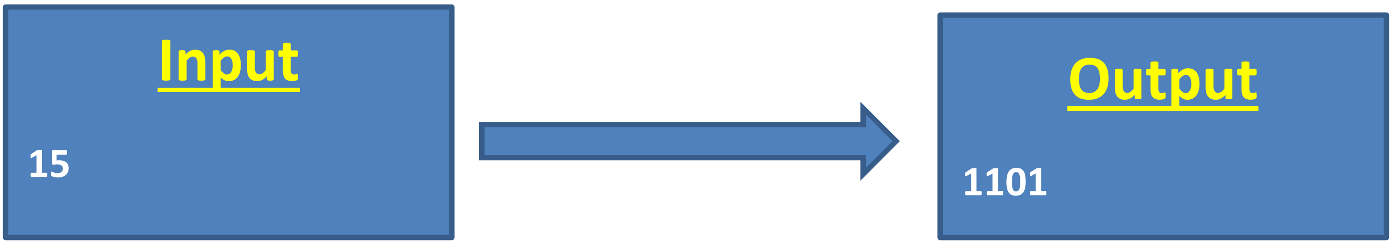
Input

7

Output

111

Example 2:



Input

15

Output

1101

Odd & even

Problem statement:

Write a program that take an integer (n) from user then calculates and prints the product of odd integers from 1 to n and the summation of even integers from 1 to n.

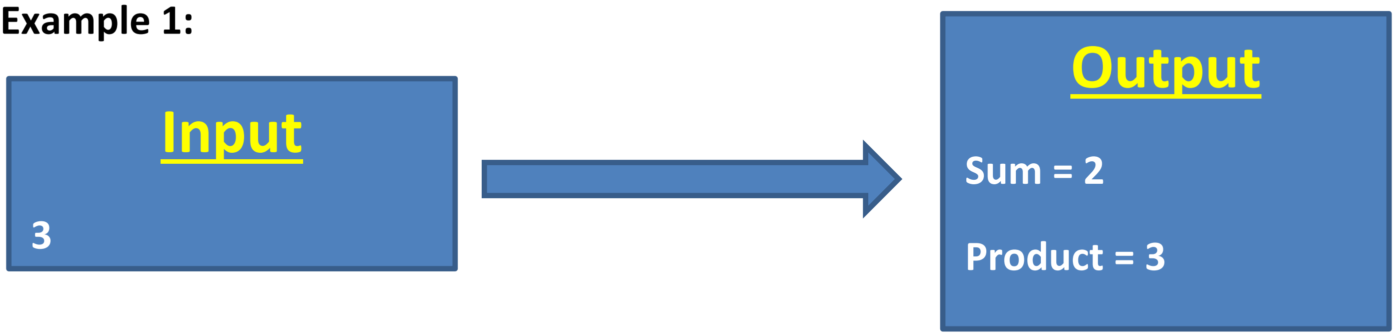
Input:

An integer (n).

Output:

The product of odd integers from 1 to n and the summation of even integers from 1 to n.

Example 1:



Input

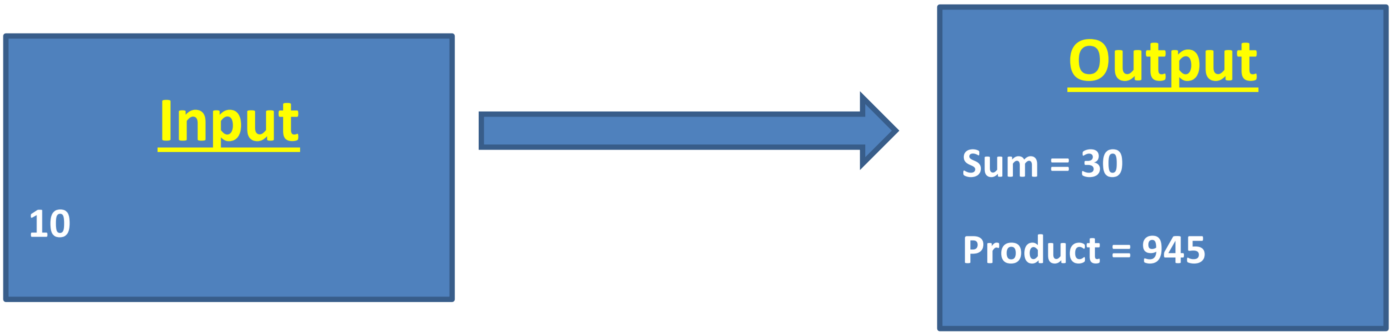
3

Output

Sum = 2

Product = 3

Example 2:



Input

10

Output

Sum = 30

Product = 945

Find the Smallest

Problem statement:

Write a program that finds the smallest of (n) several integers. The first value is read specifies (n).

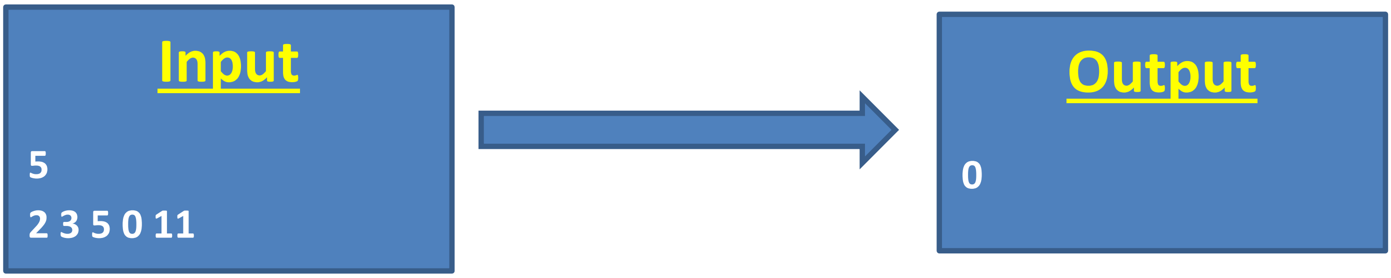
Input:

An integer (n).
Then you will take n integer numbers a_i.

Output:

Smallest of n numbers.

Example 1:



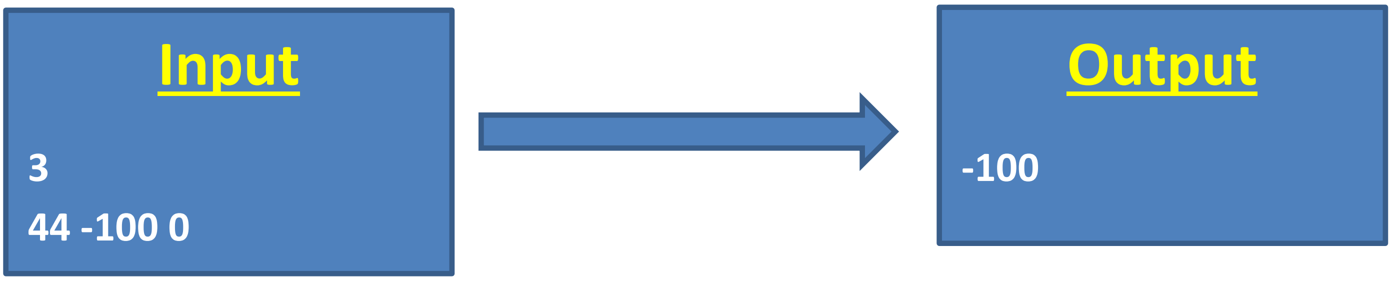
Input

5
2 3 5 0 11

Output

0

Example 2:



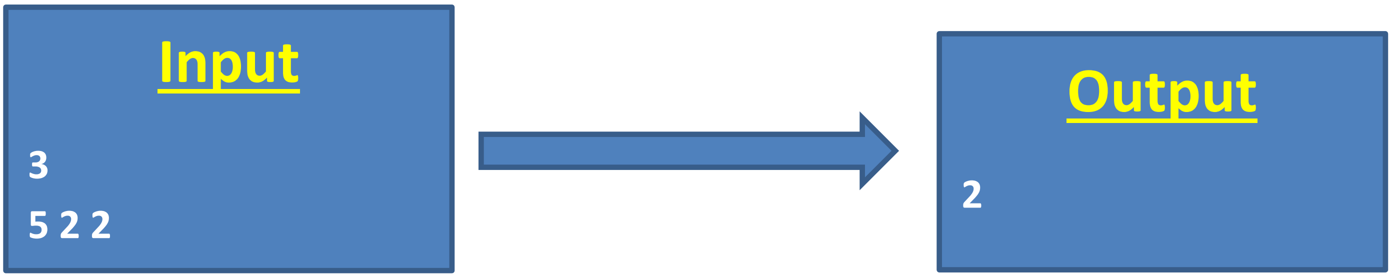
Input

3
44 -100 0

Output

-100

Example 3:



Input

3
5 2 2

Output

2

SUM Till >=200

Problem statement:

Write C program to read set of numbers and print the sum of all number (Your program will take input as long as the summation of all previously given numbers is smaller than 200 otherwise the program will print the summation then stop)

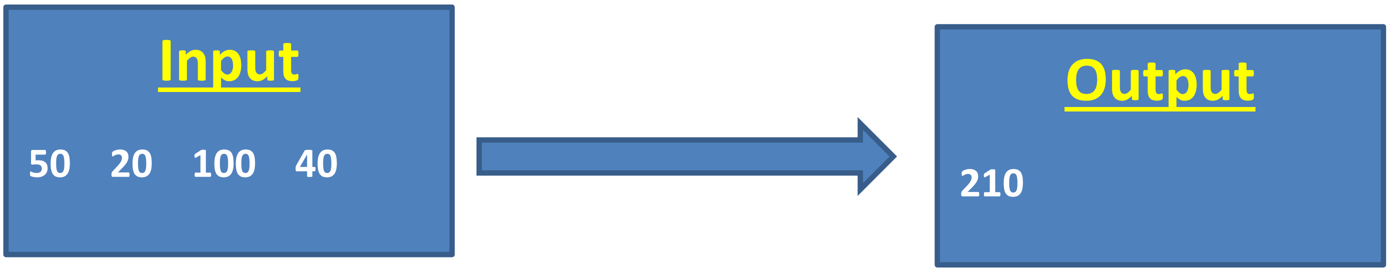
Input:

Set of number

Output:

Sum of numbers

Example 1:



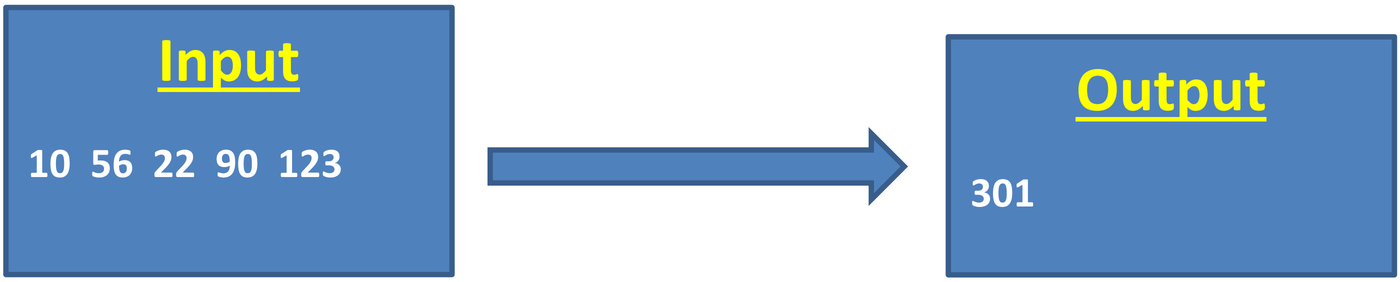
Input

50 20 100 40

Output

210

Example 2:



Input

10 56 22 90 123

Output

301

Sales Commission Calculator

Problem statement:

One large chemical company pays its salespeople on a commission basis. The salespeople receive $200 per week plus 9% of their gross sales for that week. For example, a salesperson who sells $5000 worth of chemicals in a week receives $200 plus 9% of $5000, or a total of $650. Develop a program that will input each salesperson’s gross sales for last week and will calculate and display that salesperson's earnings. Process one salesperson's figures at a time.

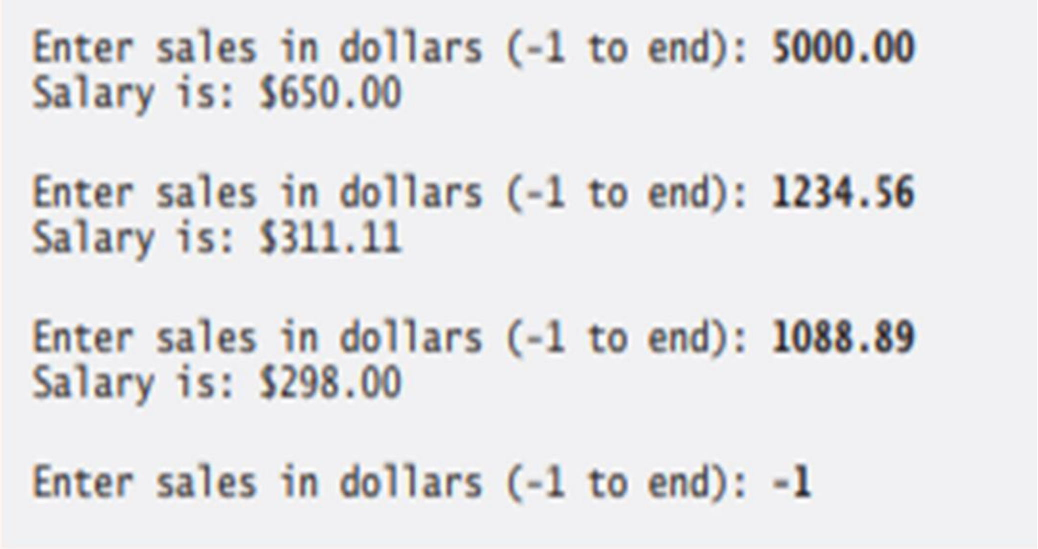
Input:

N integers.

Output:

Salary of each employee. **And your program has to be terminated when user enter (-1)**

Example 1:



```
Enter sales in dollars (-1 to end): 5000.00
Salary is: $650.00

Enter sales in dollars (-1 to end): 1234.56
Salary is: $311.11

Enter sales in dollars (-1 to end): 1088.89
Salary is: $298.00

Enter sales in dollars (-1 to end): -1
```

Salary Calculator

Problem statement:

Develop a program that will determine the gross pay for each of several employees. The company pays “straight time” for the first 40 hours worked by each employee and pays “time-and-a-half” for all hours worked in excess of 40 hours. You’re given a list of the employees of the company, the number of hours each employee worked last week and the hourly rate of each employee. Your program should input this information for each employee, and should determine and display the employee's gross pay. Process one salesperson's figures at a time.

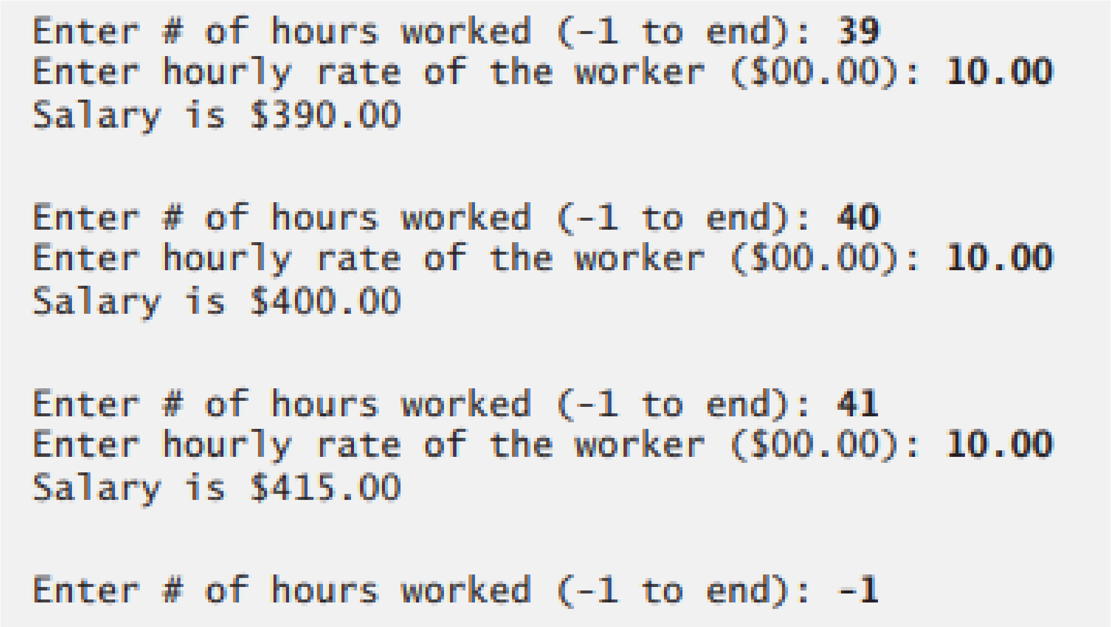
Input:

N integers.

Output:

Salary of each employee. **And your program has to be terminated when user enter (-1)**

Example 1:

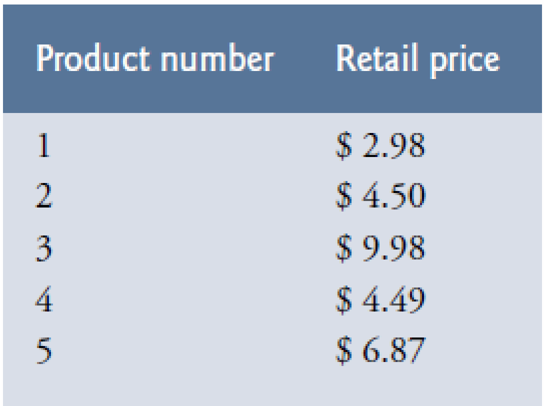


```
Enter # of hours worked (-1 to end): 39
Enter hourly rate of the worker ($00.00): 10.00
Salary is $390.00

Enter # of hours worked (-1 to end): 40
Enter hourly rate of the worker ($00.00): 10.00
Salary is $400.00

Enter # of hours worked (-1 to end): 41
Enter hourly rate of the worker ($00.00): 10.00
Salary is $415.00

Enter # of hours worked (-1 to end): -1
```

Calculating Sales

Problem statement:

An online retailer sells five different products whose retail prices are shown in the following table:



Product number	Retail price
1	$ 2.98
2	$ 4.50
3	$ 9.98
4	$ 4.49
5	$ 6.87

Write a program that reads a series of pairs of numbers as follows:

a) Product number

b) Quantity sold for one day

Your program should calculate and display the total retail value of all products sold last week.

Input:

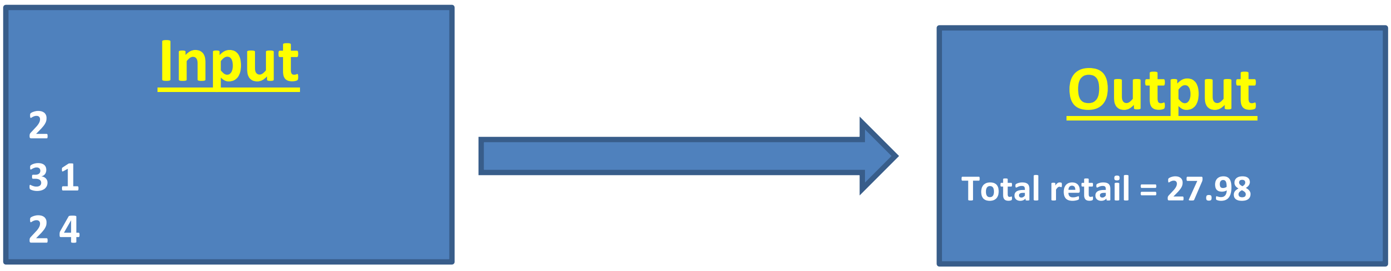
An Integer (n) number refers to number of given pairs. Each pair has to numbers:

1. Product number
2. Quantity sold for one day

Output:

Total retail value of all products.

Example 1:



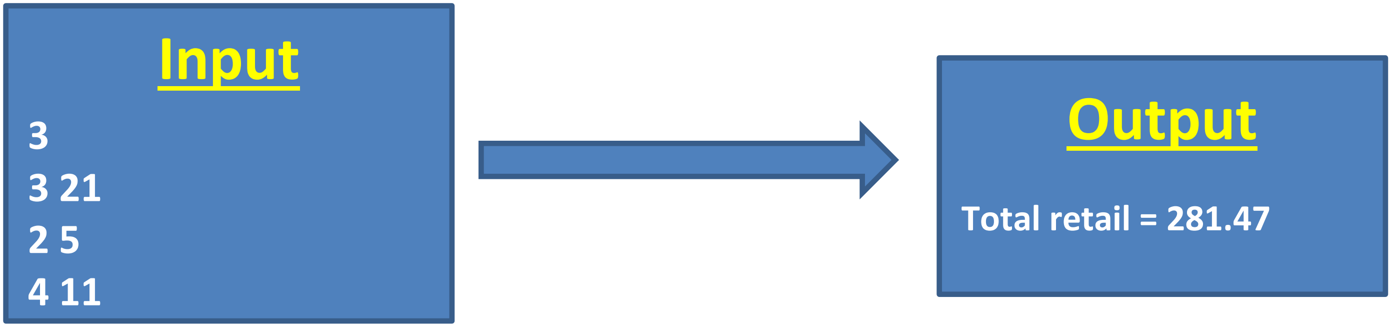
Input

```
2
3 1
2 4
```

Output

```
Total retail = 27.98
```

Example 2:



Input

```
3
3 21
2 5
4 11
```

Output

```
Total retail = 281.47
```

Factorial

Problem statement:

Write a program that prints the factorial of a given number.

Input:

An integer (x).

Output:

The factorial of **x!**

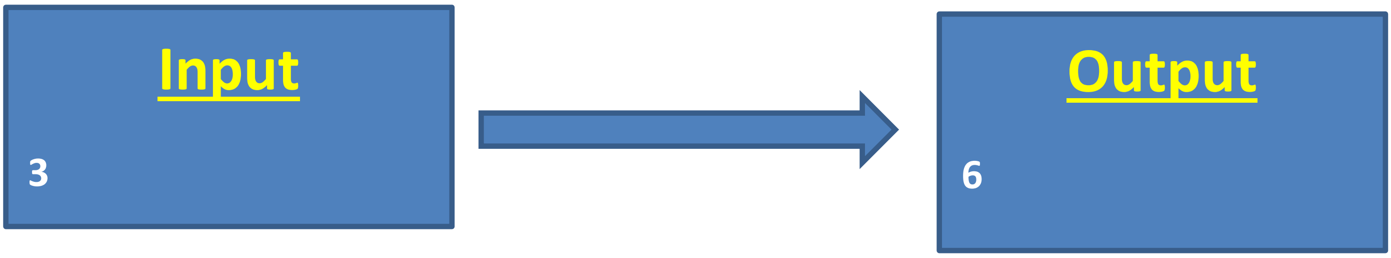
Notes:

The factorial of a number is the multiplication of numbers from 1 to that number

3! = 3*2*1 = 6

5! = 5*4*3*2*1 = 120

Example 1:



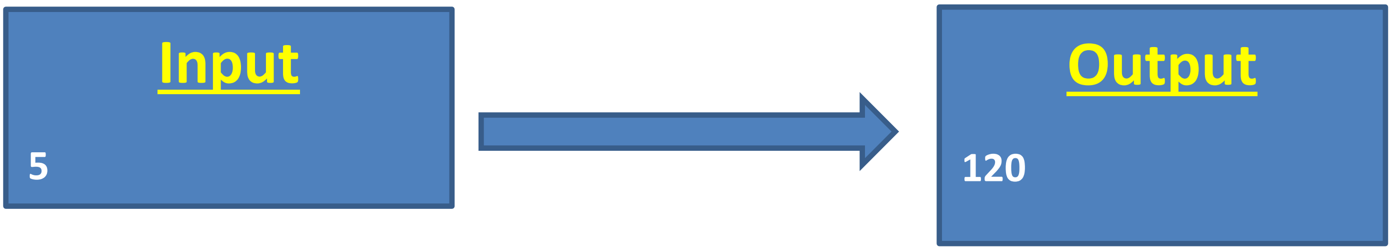
Input

3

Output

6

Example 2:



Input

5

Output

120

Table of Decimal, Binary, Octal and Hexadecimal Equivalents

Problem statement:

Write a program that prints a table of the binary, octal and hexadecimal equivalents of the decimal numbers in the range 1 through 256.

Hints:

Search for methods that convert from decimal to octal directly like this:
https://byjus.com/maths/convert-decimal-to-octal/
Search for methods that convert from decimal to Hexadecimal directly like this:
https://www.permadi.com/tutorial/numDecToHex/

Output:

Decimal	Binary	Octal	Hexadecimal
0	0	0	0
1	1	1	1
2	10	2	2
3	11	3	3
4	100	4	4
5	101	5	5
6	110	6	6
7	111	7	7
8	1000	10	8
9	1001	11	9
10	1010	12	A
11	1011	13	B
12	1100	14	C
13	1101	15	D
14	1110	16	E
15	1111	17	F
16	10000	20	10

GCD

Problem statement:

Write C program to to find the greatest common divisor of two numbers.

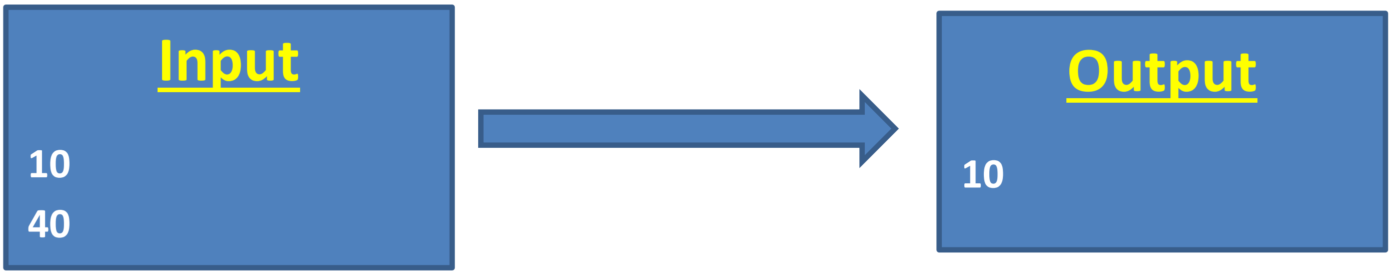
Input:

2 integers.

Output:

Greatest common divisor of two numbers.

Example 1:



Input

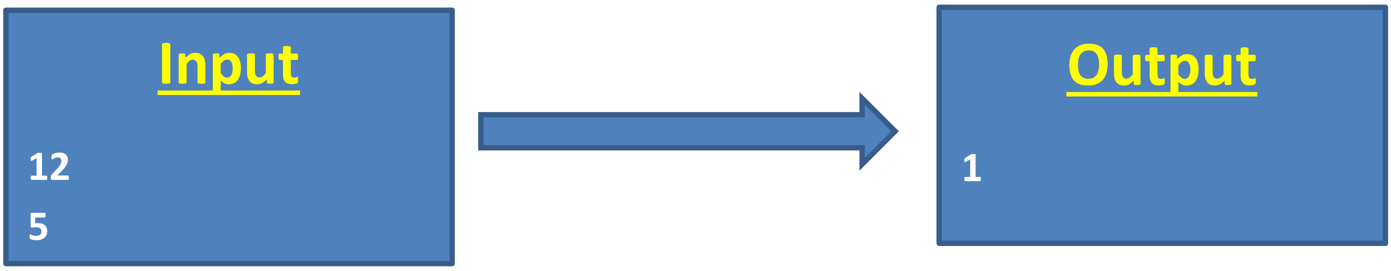
10

40

Output

10

Example 2:



Input

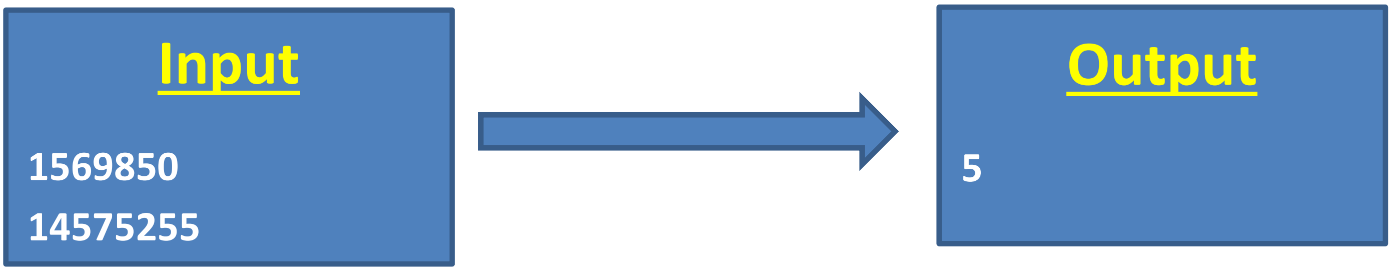
12

5

Output

1

Example 3:



Input

1569850

14575255

Output

5

Super Mario

Problem statement:

Do you know Super Mario game?

We changed the game rules, Mario instead of avoiding the monsters by jumping he will fight them all. But because he is very weak always he loses the fight and loses his health. At the beginning of the game Mario starts with health **100**, then he fights (n) monsters.

Each monster (i) has a power (x_i) and when Mario fights it, he lose and his health will be decreased by (x_i). While Mario health is a positive value he still alive otherwise he dead. You will be given the number of monsters (n) and the power of each monster (x_i) can you tell us if Mario will be alive or be dead after fighting all monsters.

Notes:

Positive value means (value > 0)

Input:

A number (n) refers to the number of monsters. Then (n) numbers

Each number (x_i) refers to monster's power.

Output:

If Mario still alive after fighting all monsters output **"Alive"** otherwise output **"Game over"**

Example 1:

Input

3
100 21 3

→

Output

Game over

Example 2:

Input

7
8 30 11 0 10 3 5

→

Output

Alive

Armstrong Numbers

Problem statement:

Write C program to print out number of all Armstrong numbers between 1 and **n**. While (n) is a given number.

Note:

If sum of cubes of each digit of the number is equal to the number itself, then the number is called an Armstrong number.

For example, 153 = (1 * 1 * 1) + (5 * 5 * 5) + (3 * 3 * 3)

Input:

An integer **n** (1=<n<=10^6)

Output:

to print out all Armstrong numbers between 1 and **n**.

Example 1:

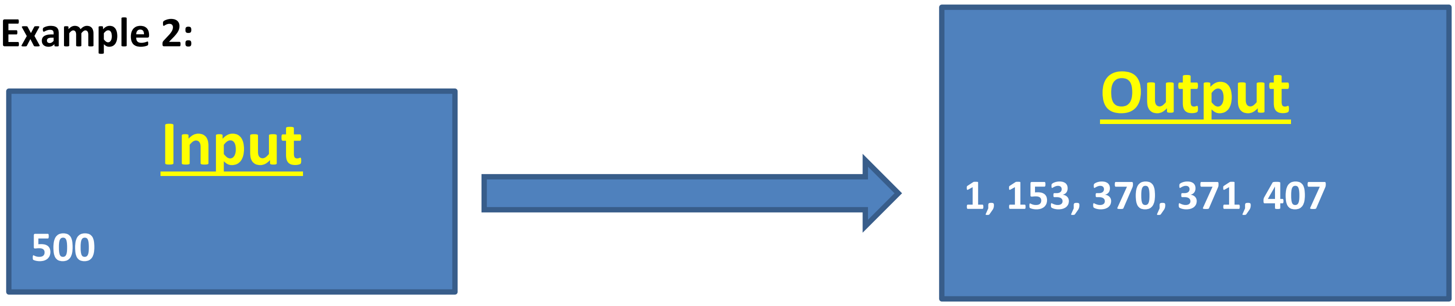
Input

10

Output

1

Example 2:



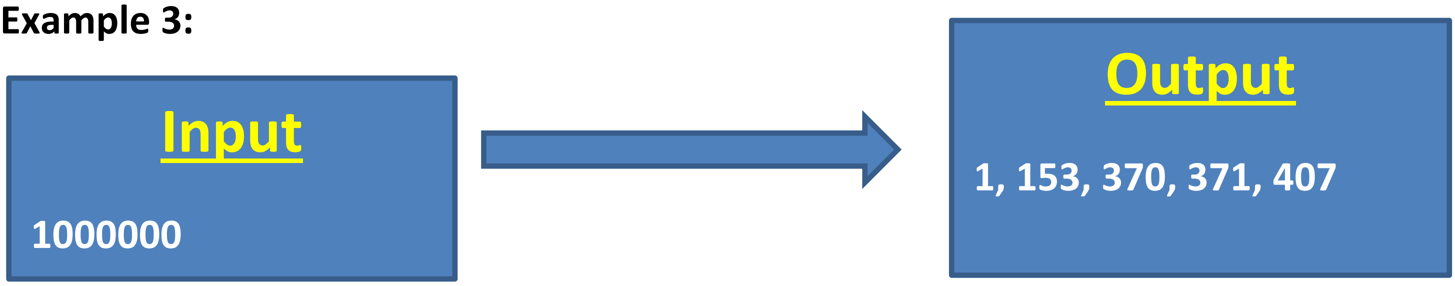
Input

500

Output

1, 153, 370, 371, 407

Example 3:



Input

1000000

Output

1, 153, 370, 371, 407

Fibonacci series

Problem statement:

Write a program to print Fibonacci series of n terms where n is input by user:

Note:

The Fibonacci sequence is the series of numbers:
$F_0 = 0$
$F_1 = 1$
$F_n = F_{n-1} + F_{n-2}$

0, 1, 1, 2, 3, 5, 8, 13, 21, 34, ...

The next number is found by adding up the two numbers before it:

- the 2 is found by adding the two numbers before it (1+1),
- the 3 is found by adding the two numbers before it (1+2),
- the 5 is (2+3),
- and so on!

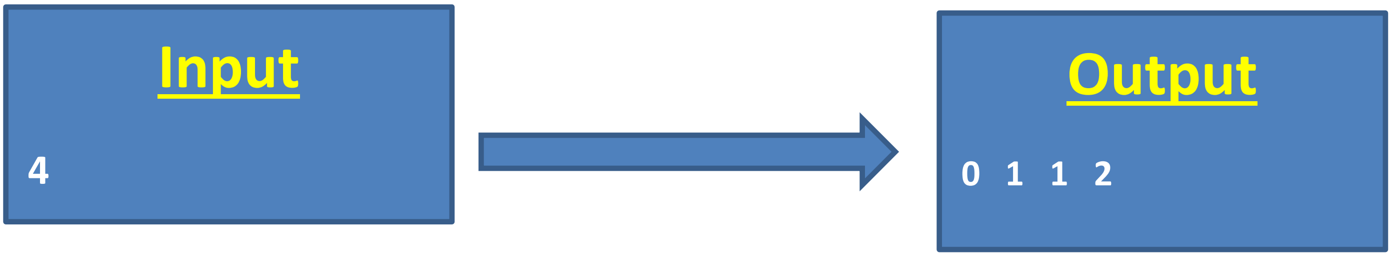
Input:

1 integer (n) of number of elements.

Output:

Fibonacci sequence with length = n. **don't use array**

Example 1:



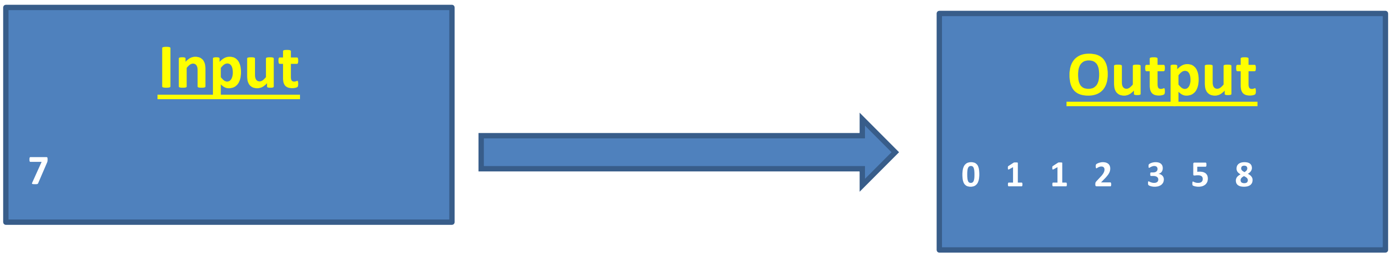
Input

4

Output

0 1 1 2

Example 2:



Input

7

Output

0 1 1 2 3 5 8