

Program-01

A Prime-Adam integer is a positive integer (without leading zeros) which is a prime as well as an Adam number.

Prime number: A number which has only two factors, i.e. 1 and the number itself.

Example: 2, 3, 5, 7 ... etc.

Adam number: The square of a number and the square of its reverse are reverse to each other.

Example: If $n = 13$ and reverse of ' n ' = 31, then,

$$(13)^2 = 169$$

$$(31)^2 = 961 \text{ which is reverse of } 169$$

thus 13, is an Adam number.

Accept two positive integers m and n , where m is less than n as user input. Display all Prime-Adam integers that are in the range between m and n (both inclusive) and output them along with the frequency, in the format given below:

Test your program with the following data and some random data:

Example 1

INPUT:

$$m = 5$$

$$n = 100$$

OUTPUT:

THE PRIME-ADAM INTEGERS ARE:

11 13 31

FREQUENCY OF PRIME-ADAM INTEGERS IS: 3

Example 2

INPUT:

$$m = 100$$

$$n = 200$$

OUTPUT:
THE PRIME-ADAM INTEGERS ARE:
101 103 113
FREQUENCY OF PRIME-ADAM INTEGERS IS: 3

Example 3

INPUT:
m = 50
n = 70

OUTPUT:
THE PRIME-ADAM INTEGERS ARE:
NIL
FREQUENCY OF PRIME-ADAM INTEGERS IS: 0

Example 4

INPUT:
m = 700
n = 450

OUTPUT:
INVALID INPUT

ALGORITHM

- Step-1 :- START
- Step-2 :- Create a class named as prime_adam.
- Step-3 :- Create a function named as isPrime which takes an integer as an argument and returns a boolean value. In this function, first initialize the counter to 0 and then start a for loop (from 1 to the given number) and check if the given number is divisible by the current number in the loop. If it is divisible, then increment the counter by 1. If the counter is greater than 2, then return False. Else, return True.
- Step-4 :- Create a function named as reverse which takes an integer as an argument and returns an integer. In this function, first initialize the variable rev to 0. Then, start a while loop and in each iteration, multiply the rev by 10 and add the remainder of the given number divided by 10 to it. Then, divide the given number by 10. Repeat this until the given number becomes 0. Finally, return the rev.
- Step-5 :- Create a function named as isAdam which takes an integer as an argument and returns a boolean value. In this function, first check if the given number is prime or not (by calling isPrime function). If it is not prime, then return False. Else, check if the reverse of the square of the given number (by calling reverse function) is equal to the square of the reverse of the given number. If it is equal, then return True. Else, return False.
- Step-6 :- Create a function named as main to call the methods and print the result. In this function, first initialize the variable m and n using Scanner Class. Then, start a for loop (from m to n) and check if the current number in the loop is an Adam number or not by calling the isAdam function. If it is an Adam number, then print it.
- Step-7 :- END

VD TABLE

Sr. No.	Variable	Data Type	Description
1	x	int	To store the number
2	r	int	To store the reverse of the number
3	s1	int	To store the square of the number
4	s2	int	To store the square of the reverse of the number
5	i	int	To store the value of the loop variable
6	c	int	Used As Counter Variable
7	count	int	Used As Counter Variable
8	m	int	To store the upper limit for the loop
9	n	int	To store the lower limit for the loop

OUTPUT

```
BlueJ: Terminal Window - ISC
Options
Enter the value of m
5
Enter the value of n
100
Prime Adam Numbers Are:
11
13
31
Frequency: 3
Can only enter input while your program is running
```

Program-02

Write a program to declare a matrix $A[][]$ of order (M x N) where 'M' is the number of rows and 'N' is the number of columns such that the value of 'M' must be greater than 0 and less than 10 and the value of 'N' must be greater than 2 and less than 6. Allow the user to input digits (0 - 7) only at each location, such that each row represents an octal number.

Example:

2	3	1	(decimal equivalent of 1st row = 153 i.e. $2 \times 8^2 + 3 \times 8^1 + 1 \times 8^0$)
4	0	5	(decimal equivalent of 2nd row = 261 i.e. $4 \times 8^2 + 0 \times 8^1 + 5 \times 8^0$)
1	5	6	(decimal equivalent of 3rd row = 110 i.e. $1 \times 8^2 + 5 \times 8^1 + 6 \times 8^0$)

Perform the following tasks on the matrix:

Display the original matrix.

Calculate the decimal equivalent for each row and display as per the format given below.

Test your program for the following data and some random data:

Example 1:

INPUT:

M = 1

N = 3

ENTER ELEMENTS FOR ROW 1: 1 4 4

OUTPUT:

FILLED MATRIX			DECIMAL EQUIVALENT
1	4	4	100

Example 2:

INPUT:

M = 3

N = 4

ENTER ELEMENTS FOR ROW 1: 1 1 3 7

ENTER ELEMENTS FOR ROW 2: 2 1 0 6

ENTER ELEMENTS FOR ROW 3: 0 2 4 5

OUTPUT:

FILLED MATRIX				DECIMAL EQUIVALENT
1	1	3	7	607

2	1	0	6	1094
0	2	4	5	165

Example 3:

INPUT:

M = 3

N = 3

ENTER ELEMENTS FOR ROW 1: 2 4 8

OUTPUT:

INVALID INPUT

Example 4:

INPUT:

M = 4

N = 6

OUTPUT:

OUT OF RANGE

ALGORITHM

- Step-1 :- START
- Step-2 :- Create a class named as decimal.
- Step-3 :- Create a function named as dec_con which takes an integer type array and two integer arguments and displays the decimal equivalent of the given number. In this function, first create a for loop (from 0 to row length of array) inside which create a variable named decNum and initialise it with 0. Inside the running for loop, start another for loop (from 0 to column length of array) and then store the sum value of the the array indexes at [i][j] raised to the power (8,n-j-i) in the variable decNum, in this loop print the array element at [i][j]. After the inner for loop ends, print the value of decNum.
- Step-4 :- Create a function named as main and call the method dec_con and pass the array and the number of rows and columns as arguments after taking the input of number of rows and columns and the array from the user.
- Step-5 :- END

VD TABLE

Sr. No.	Variable	Data Type	Description
1	i	int	To store the value of the loop variable
2	j	int	To store the value of the loop variable
3	m	int	To store the number of rows
4	n	int	To store the number of columns
5	decNum	int	To store the sum of decimal equivalent of the array indexes
6	a	int	To store the array elements

OUTPUT

```
BlueJ: Terminal Window - basic
Options
Enter the number of rows (M): 3
Enter the number of columns (N): 4
ENTER ELEMENTS FOR ROW 1:
1 1 3 7
ENTER ELEMENTS FOR ROW 2:
2 1 0 6
ENTER ELEMENTS FOR ROW 3:
0 2 4 5
FILLED MATRIX    DECIMAL EQUIVALENT
1 1 3 7          607
2 1 0 6          1094
0 2 4 5          165
```

Program-03

Write a program to accept a sentence which may be terminated by either '.', '?' or '!' only. The words are to be separated by a single blank space and are in UPPER CASE.

Perform the following tasks:

- Check for the validity of the accepted sentence only for the terminating character.

- Arrange the words in ascending order of their length. If two or more words have the same length, then sort them alphabetically.

- Display the original sentence along with the converted sentence.

Test your program for the following data and some random data:

Example 1:

INPUT:

AS YOU SOW SO SHALL YOU REAP.

OUTPUT:

AS YOU SOW SO SHALL YOU REAP.

AS SO SOW YOU YOU REAP SHALL

Example 2:

INPUT:

SELF HELP IS THE BEST HELP.

OUTPUT:

SELF HELP IS THE BEST HELP.

IS THE BEST HELP HELP SELF

Example 3:

INPUT:

BE KIND TO OTHERS.

OUTPUT:

BE KIND TO OTHERS.

BE TO KIND OTHERS

Example 4:

INPUT:
NOTHING IS IMPOSSIBLE#

OUTPUT:
INVALID INPUT

ALGORITHM

- Step-1 :- START
- Step-2 :- Create a class named as check.
- Step-3 :- Create a function named as sen_check and pass the string type parameter sen. In this function, create a string tokenizer object and pass the string sen and the delimiter as ? . ! , . Create a variable count to store the number of tokens. Create a string type array a[] and now using a for loop (from 0 to count) and store the tokens in the string type array a[]. Create a for loop (from 0 to the length of the array), start a inner loop (from 0 to a.length-1-i) and check whether a[j].compareTo(a[j+1])>0 is true then swap the adjacent elements. Create a for loop (from 0 to the length of the array), start a inner loop (from 0 to a.length-1-i) and check whether a[j].length() > a[j + 1].length(), if true then swap them if they are not in the desired order. Now print the sorted array.
- Step-4 :- Create a function named as main. In this function, create a string type variable sen and store the sentence in it (user input). Call the function sen_check and pass the string sen as the parameter.
- Step-5 :- END

VD TABLE

Sr. No.	Variable	Data Type	Description
1	i	int	To store the value of the loop variable
2	j	int	To store the value of the loop variable
3	count	int	To store the number of tokens
4	a[]	String	To store the tokens
5	sen	String	To store the sentence
6	temp	String	To store the temporary value
7	last	char	To store the last character of the sentence
8	len	int	To store the length of the sentence

OUTPUT

```
BlueJ: Terminal Window - basic
Options
INPUT:
AS YOU SOW SO SHALL YOU REAP.
OUTPUT:
AS YOU SOW SO SHALL YOU REAP.
AS SO SOW YOU YOU REAP SHALL
```

Program-04

Design a program to accept a day number (between 1 and 366), year (in 4 digits) from the user to generate and display the corresponding date. Also, accept 'N' ($1 \leq N \leq 100$) from the user to compute and display the future date corresponding to 'N' days after the generated date. Display an error message if the value of the day number, year and N are not within the limit or not according to the condition specified.

Test your program with the following data and some random data:

Example 1

INPUT:

DAY NUMBER: 255

YEAR: 2018

DATE AFTER (N DAYS): 22

OUTPUT:

DATE: 12TH SEPTEMBER, 2018

DATE AFTER 22 DAYS: 4TH OCTOBER, 2018

Example 2

INPUT:

DAY NUMBER: 360

YEAR: 2018

DATE AFTER (N DAYS): 45

OUTPUT:

DATE: 26TH DECEMBER, 2018

DATE AFTER 45 DAYS: 9TH FEBRUARY, 2019

Example 3

INPUT:

DAY NUMBER: 500

YEAR: 2018

DATE AFTER (N DAYS): 33

OUTPUT:

DAY NUMBER OUT OF RANGE

Example 4

INPUT:

DAY NUMBER: 150

YEAR: 2018

DATE AFTER (N DAYS): 330

OUTPUT:

DATE AFTER (N DAYS) OUT OF RANGE

ALGORITHM

- Step-1 :- START
- Step-2 :- Create a class named as days.
- Step-3 :- Create a method named as main and declare the Scanner class. Declare the variables- n,day,year- of integer type to store the number of days, calculated days and year respectively. Create an array named a of integer type in order to store the names of the 12 months orderwise. Check whether the year entered is a leap year or not, if true then store the number of days in the month of February as 29 else store it as 28. Using multiple for-loops store the calculated days in the variable day and the year in the variable year. Finally print the calculated date,month and y
- Step-4 :- END

VD TABLE

Sr. No.	Variable	Data Type	Description
1	n	int	To store the number of days
2	day	int	To store the days
3	year	int	To store the year
4	a	int[]	To store the names of the 12 months orderwise
5	i	int	Used in for-loop
6	j	int	Used in for-loop
7	k	int	Used in for-loop
8	nday	int	To store the calculated day number
9	date	int	To store the calculated date

OUTPUT

```
BlueJ: Terminal Window - basic
Options
DAY NUMBER: 255
YEAR: 2018
DATE AFTER (N DAYS): 22
DATE : 12th September 2018
DATE AFTER 22 DAYS: 4th October 2018
```

Program-05

Write a program to declare a single-dimensional array `a[]` and a square matrix `b[][]` of size `N`, where $N > 2$ and $N < 10$. Allow the user to input positive integers into the single dimensional array.

Perform the following tasks on the matrix:

Sort the elements of the single-dimensional array in ascending order using any standard sorting technique and display the sorted elements.

Fill the square matrix `b[][]` in the following format:

If the array `a[] = {5, 2, 8, 1}` then, after sorting `a[] = {1, 2, 5, 8}`

Then, the matrix `b[][]` would fill as below:

```
1 2 5 8
1 2 5 1
1 2 1 2
1 1 2 1
```

Display the filled matrix in the above format.

Test your program for the following data and some random data:

Example 1

INPUT:

`N = 3`

ENTER ELEMENTS OF SINGLE DIMENSIONAL ARRAY: `3 1 7`

OUTPUT:

SORTED ARRAY: `1 3 7`

FILLED MATRIX

```
1 3 7
1 3 1
1 1 3
```

Example 2

INPUT:

`N = 13`

OUTPUT:
MATRIX SIZE OUT OF RANGE

Example 3

INPUT:
N = 5
ENTER ELEMENTS OF SINGLE DIMENSIONAL ARRAY: 10 2 5 23 6

OUTPUT:
SORTED ARRAY: 2 5 6 10 23
FILLED MATRIX

2 5 6 10 23
2 5 6 10 2
2 5 6 2 5
2 5 2 5 6
2 2 5 6 10

ALGORITHM

- Step-1 :- START
- Step-2 :- Create a class named as sort.
- Step-3 :- Create a method named as sort and pass the integer type array a[] as parameter. In this function, create a for loop (from 0 to the length of the array), start a inner loop (from 0 to a.length-1-i) and check whether a[j] > a[j + 1], if true then swap them if they are not in the desired order.
- Step-4 :- Create a method named as main. In this function, create an integer type array a[] and store the elements in it (user input). Call the function sort and pass the array a[] as the parameter. Print the sorted array. Create a 2-Dimensional array b[][] and store the elements of sorted array a[] in the required pattern using for-loops. Now print the 2-Dimensional array b[][].
- Step-5 :- END

VD TABLE

Sr. No.	Variable	Data Type	Description
1	i	int	To store the value of the loop variable
2	j	int	To store the value of the loop variable
3	a	int[]	To store the elements of the array
4	b	int[][]	To store the elements of the array in the required pattern
5	n	int	To store the size of the array
6	temp	int	To store the temporary value
7	k	int	To store the value of the loop variable
8	r	int	To store the value of the loop variable

OUTPUT

```
BlueJ: Terminal Window - basic
Options
ENTER VALUE OF N: 3
ENTER ELEMENTS OF SINGLE DIMENSIONAL ARRAY:
3 1 7
SORTED ARRAY:
1 3 7
FILLED MATRIX:
1 3 7
1 3 1
1 1 3
```

Program-06

Write a program to accept a sentence which may be terminated by either . , ? or ! only. The words are to be separated by a single blank space and are in uppercase.

Perform the following tasks:

(a) Check for the validity of the accepted sentence.

(b) Convert the non-palindrome words of the sentence into palindrome words by concatenating the word by its reverse (excluding the last character).

Example:

The reverse of the word HELP would be LEH (omitting the last alphabet) and by concatenating both, the new palindrome word is HELPLEH. Thus, the word HELP becomes HELPLEH.

Note: The words which end with repeated alphabets, for example ABB would become ABBA and not ABBBA and XAZZZ becomes XAZZZAX.

[Palindrome word: Spells same from either side. Example: DAD, MADAM etc.]

(c) Display the original sentence along with the converted sentence.

Test your program for the following data and some random data:

Example 1

INPUT:

THE BIRD IS FLYING.

OUTPUT:

THE BIRD IS FLYING.

THEHT BIRDRIb ISI FLYINGNIYLF

Example 2

INPUT:

IS THE WATER LEVEL RISING?

OUTPUT:

IS THE WATER LEVEL RISING?
ISI THEHT WATERETAW LEVEL RISINGNISIR

Example 3

INPUT:
THIS MOBILE APP LOOKS FINE.

OUTPUT:
THIS MOBILE APP LOOKS FINE.
THISIHT MOBILELIBOM APPA LOOKSKOOL FINENIF

Example 3

INPUT:
YOU MUST BE CRAZY#

OUTPUT:
INVALID INPUT

ALGORITHM

- Step-1 :- START
- Step-2 :- Create a class named as pallin.
- Step-3 :- Create a method named as isPallindrome and pass a string named word as a parameter. In this function check whether the string is pallindrome or not.
- Step-4 :- Create a method named as makePalindrome and pass a string named word as a parameter. In this function, declare variables to store length and the last character of the string word in len and lastChar respectively. Using a StringBuffer class and a for-loop, append the string word to the StringBuffer object sb and then append the last character of the string word to the StringBuffer object sb. Then, return the StringBuffer object sb as a string.
- Step-5 :- Create a method named as main. In this function, using Scanner class take the input of the string Str as input. Then, call the method isPallindrome and pass the string word as a parameter. If the method returns true, then print the string Str. Then, call the method makePalindrome and pass the string word as a parameter. Then, print the string returned by the method makePalindrome.
- Step-6 :- END

VD TABLE

Sr. No.	Variable	Data Type	Description
1	Str	String	To store the input string
2	word	String	To store the words of the string
3	str	String	To store the string without front and back spaces
4	len	int	To store the length of the string
5	lastChar	char	To store the last character of the string
6	i	int	Used in for-loop
7	isPalindrome	boolean	To check whether the word is pallindrome or not
8	palin	boolean	To check whether the word is pallindrome or not
9	palinWord	String	To store the string returned by the method makePalindrome
10	convertedStr	String	To store the converted string

OUTPUT

```
BlueJ: Terminal Window - basic
Options
ENTER THE SENTENCE:
THE BIRD IS FLYING.

THE BIRD IS FLYING.
THEHT BIRDRI8 ISI FLYINGNIYLF
```

Program-07

A Goldbach number is a positive even integer that can be expressed as the sum of two odd primes.

Note: All even integer numbers greater than 4 are Goldbach numbers.

Example:

$$6 = 3 + 3$$

$$10 = 3 + 7$$

$$10 = 5 + 5$$

Hence, 6 has one odd prime pair 3 and 3. Similarly, 10 has two odd prime pairs, i.e. 3 and 7, 5 and 5.

Write a program to accept an even integer 'N' where $N > 9$ and $N < 50$. Find all the odd prime pairs whose sum is equal to the number 'N'.

Test your program with the following data and some random data:

Example 1

INPUT:

N = 14

OUTPUT:

PRIME PAIRS ARE:

3, 11

7, 7

Example 2

INPUT:

N = 30

OUTPUT:

PRIME PAIRS ARE:

7, 23

11, 19

13, 17

Example 3

INPUT:

N = 17

OUTPUT:

INVALID INPUT. NUMBER IS ODD.

Example 4

INPUT:

N = 126

OUTPUT:

INVALID INPUT. NUMBER OUT OF RANGE.

ALGORITHM

- Step-1 :- START
- Step-2 :- Create a class named as goldbach.
- Step-3 :- Create a method named as even of type boolean and pass an integer named n as a parameter. In this function, check whether the number is even or not.
- Step-3 :- Create a method named as odd_prime of type boolean and pass an integer named n as a parameter. In this function, first call the function even and check whether the number is even or not. If the number is not even then check whether the number is prime or not.
- Step-4 :- Create a method named as main. In this function, first take an integer input from the user and store it in a variable named n. Now call the function even and check the required, then call the function odd_prime and check whether the number is prime or not. Create two for-loops (from 1 to n and from 1 to i(variable of the outer loop)), now check whether the both the loop variables is prime or not by calling the function odd_prime and passing the loop variable as parameter. If the number is prime then print the number.
- Step-5 :- END

VD TABLE

Sr. No.	Variable	Data Type	Description
1	n	int	To store the input number
2	i	int	To store the value of the outer loop
3	j	int	To store the value of the inner loop
4	k	int	Used as counter variable
5	c	int	Used as counter variable

OUTPUT

```
BlueJ: Terminal Window - basic
Options
INPUT:
N= 14

OUTPUT:
PRIME PAIRS ARE:
3, 11
7, 7
11, 3
```

Program-08

Write a program to declare a matrix `a[][]` of order $(m \times n)$ where 'm' is the number of rows and 'n' is the number of columns such that the values of both 'm' and 'n' must be greater than 2 and less than 10. Allow the user to input integers into this matrix. Perform the following tasks on the matrix:

Display the original matrix.

Sort each row of the matrix in ascending order using any standard sorting technique.

Display the changed matrix after sorting each row.

Test your program for the following data and some random data:

Example 1

INPUT:

M = 4

N = 3

ENTER ELEMENTS OF MATRIX:

11 -2 3

5 16 7

9 0 4

3 1 8

OUTPUT:

ORIGINAL MATRIX

11 -2 3

5 16 7

9 0 4

3 1 8

MATRIX AFTER SORTING ROWS

-2 3 11

5 7 16

0 4 9

1 3 8

Example 2

INPUT:

M = 3

N = 3

ENTER ELEMENTS OF MATRIX

22 5 19

7 36 12

9 13 6

OUTPUT:

ORIGINAL MATRIX

22 5 19

7 36 12

9 13 6

MATRIX AFTER SORTING ROWS

5 19 22

7 12 36

6 9 13

Example 3

INPUT:

M = 11

N = 5

OUTPUT:

MATRIX SIZE OUT OF RANGE.

ALGORITHM

- Step-1 :- START
- Step-2 :- Create a class named as sort.
- Step-3 :- Create a method named as main. In this function, create variables named as m and n to store the number of rows and columns of the matrix respectively. Now check whether number of rows and columns are valid or not. If not, then print Invalid Input and terminate the program. Otherwise, continue. Create a 2D array named as a[] [] of size m and n and using for loops take the array input. Now print the original array. Now create two for loops to traverse the array. In the inner loop, check whether the current element is greater than the next element or not. If yes, then swap the elements. Now print the sorted array.
- Step-4 :- END

VD TABLE

Sr. No.	Variable	Data Type	Description
1	m	int	To store the number of rows of the matrix
2	n	int	To store the number of columns of the matrix
3	arr	int[] []	To store the elements of the matrix
4	i	int	To store the value of the current row
5	j	int	To store the value of the current column
6	t	int	To store the value of the current element

OUTPUT

```
BlueJ: Terminal Window - basic
Options
ENTER THE VALUE OF M: 4
ENTER THE VALUE OF N: 3
ENTER ELEMENTS OF MATRIX:
ENTER ELEMENTS OF ROW 1:
11 -2 3
ENTER ELEMENTS OF ROW 2:
5 16 7
ENTER ELEMENTS OF ROW 3:
9 0 4
ENTER ELEMENTS OF ROW 4:
3 1 8
ORIGINAL MATRIX
11 -2 3
5 16 7
9 0 4
3 1 8
MATRIX AFTER SORTING ROWS
-2 3 11
5 7 16
0 4 9
1 3 8
```


Program-09

The names of the teams participating in a competition should be displayed on a banner vertically, to accommodate as many teams as possible in a single banner. Design a program to accept the names of N teams, where $2 < N < 9$ and display them in vertical order, side by side with a horizontal tab (i.e. eight spaces).

Test your program for the following data and some random data:

Example 1

INPUT:

N = 3

Team 1: Emus

Team 2: Road Rols

Team 3: Coyote

OUTPUT:

```
E   R   C
m   o   o
u   a   y
s   d   o
      t
    R   e
    o
    l
    s
```

Example 2

INPUT:

N = 4

Team 1: Royal

Team 2: Mars

Team 3: De Rose

Team 4: Kings

OUTPUT:

```
R   M   D   K
o   a   e   i
y   r       n
a   s   R   g
```

l o s
 s
 e

Example 3

INPUT:

N = 10

OUTPUT:

INVALID INPUT

ALGORITHM

- Step-1 :- START
- Step-2 :- Create a class named as banner.
- Step-3 :- Create a method named as main. In this function, create a variable named as n to store the value of the numbers of team participating. Create an array of type String named as teams of size n, create another variable of type int named as highLen and initialize it to 0. Now using the for loop take the input of the names of the teams and store it in the array teams, after each input check if the length of the input is greater than the value of highLen if yes then update the value of highLen to the length of the input. Now create a nested for loop and within which compare the variable of loop i with len (which stores the length of the string at the current index), if true print a blank line else print the letter of the string at the current index. Repeat the process until the loop ends.
- Step-4 :- END

VD TABLE

Sr. No.	Variable	Data Type	Description
1	n	int	To store the value of the numbers of team participating.
2	teams	String[]	To store the names of the teams.
3	highLen	int	To store the length of the longest string.
4	len	int	To store the length of the string at the current index.
5	i	int	Used in for-loop
6	j	int	Used in for-loop

OUTPUT

```
BlueJ: Test - Mozilla/5.0 - basic
Options
ENTER THE VALUE OF N: 3
Team 1: Emus
Team 2: Road Rols
Team 3: Coyote
E   R   C
m   o   o
u   a   y
s   d   o
      t
      R   e
      o
      l
      s
```

Program-10

A company manufactures packing cartons in four sizes, i.e. cartons to accommodate 6 boxes, 12 boxes, 24 boxes and 48 boxes. Design a program to accept the number of boxes to be packed (N) by the user (maximum up to 1000 boxes) and display the break-up of the cartons used in descending order of capacity (i.e. preference should be given to the highest capacity available, and if boxes left are less than 6, an extra carton of capacity 6 should be used.)

Test your program with the following data and some random data:

Example 1

INPUT:

N = 726

OUTPUT:

$48 * 15 = 720$

$6 * 1 = 6$

Remaining boxes = 0

Total number of boxes = 726

Total number of cartons = 16

Example 2

INPUT:

N = 140

OUTPUT:

$48 * 2 = 96$

$24 * 1 = 24$

$12 * 1 = 12$

$6 * 1 = 6$

Remaining boxes = $2 * 1 = 2$

Total number of boxes = 140

Total number of cartons = 6

Example 3

INPUT:

N = 4296

OUTPUT:
INVALID INPUT

ALGORITHM

- Step-1 :- START
- Step-2 :- Create a class named as boxes.
- Step-3 :- Create a method named as main. In this function, create a variable named as n to store the number of boxes. Check if the number of boxes is greater than 0 and less than 1000 or not, if true then print Invalid Input and exit the program. Create an array named as cartonSizes to store the various sizes of the boxes. Create a for-loop (from 0 to length of cartonSizes) inside which calculate and print the volume of each box size. Print the total volume of all the boxes.
- Step-4 :- END

VD TABLE

Sr. No.	Variable	Data Type	Description
1	n	int	To store the number of boxes
2	cartonSizes	int[]	To store the various sizes of the boxes
3	i	int	To store the index of the array
4	cartonCount	int	To store the volume of each box size
5	total	int	To store the total number of all the boxes
6	t	int	Used as temporary variable

OUTPUT

```
BlueJ: Terminal Window - basic
Options
Enter number of boxes (N): 726
48 * 15 = 720
6 * 1 = 6
Remaining boxes = 0
Total number of boxes = 726
Total number of cartons = 16
```

Program-11

The result of a quiz competition is to be prepared as follows:

The quiz has five questions with four multiple choices (A, B, C, D), with each question carrying 1 mark for the correct answer. Design a program to accept the number of participants N such that N must be greater than 3 and less than 11. Create a double-dimensional array of size (Nx5) to store the answers of each participant row-wise. Calculate the marks for each participant by matching the correct answer stored in a single-dimensional array of size 5. Display the scores for each participant and also the participant(s) having the highest score.

Example: If the value of N = 4, then the array would be:

	Q1	Q2	Q3	Q4	Q5
Participant 1	A	B	B	C	A
Participant 2	D	A	D	C	B
Participant 3	A	A	B	A	C
Participant 4	D	C	C	A	B
Key to the question:	D	C	C	B	A

Note: Array entries are line fed (i.e. one entry per line)

Test your program for the following data and some random data.

Example 1

INPUT:

N = 5

Participant 1 D A B C C

Participant 2 A A D C B

Participant 3 B A C D B

Participant 4 D A D C B

Participant 5 B C A D D

Key: B C D A A

OUTPUT:

Scores:

Participant 1 = 0

Participant 2 = 1

Participant 3 = 1

Participant 4 = 1

Participant 5 = 2

Highest Score:
Participant 5

Example 2

INPUT:

N = 4

Participant 1 A C C B D

Participant 2 B C A A C

Participant 3 B C B A A

Participant 4 C C D D B

Key: A C D B B

OUTPUT:

Scores:

Participant 1 = 3

Participant 2 = 1

Participant 3 = 1

Participant 4 = 3

Highest Score:

Participant 1

Participant 4

Example 3

INPUT:

N = 12

OUTPUT:

INPUT SIZE OUT OF RANGE.

ALGORITHM

- Step-1 :- START
- Step-2 :- Create a class named as quiz.
- Step-3 :- Create a method named as main. In this function, declare a Scanner variable named as sc and initialize it with new Scanner(System.in). Now using the sc variable, take the input of the number of participants in the quiz in a variable named as n. Create two arrays named as answers[] [] and key[] of size n and of type char respectively. Using two for loops (from 0 to n-1), take the input of the answers of the participants in the answers[] [] array and the correct answers in the key[] array. Now, using two for loops (from 0 to n-1), compare the answers of the participants with the correct answers storing them in an array score[] and print the number of correct answers of each participant. Now, compare each score with the maximum score and print the number of participants who have scored the maximum score. Now Print the number of participants who have scored the maximum score.
- Step-4 :- END

VD TABLE

Sr. No.	Variable	Data Type	Description
1	n	int	Number of participants in the quiz
2	answers[] []	char	Array to store the answers of the participants
3	key[]	char	Array to store the correct answers
4	score[]	int	Array to store the number of correct answers of each participant
5	hscore	int	Variable to store the maximum score
6	i	int	To store the index of the array
7	j	int	To store the index of the array

OUTPUT

```
BlueJ: Terminal Window - basic
Options
Enter the Number of Participants (N): 5
Enter answers of participants
Participant 1
D A B C C
Participant 2
A A D C B
Participant 3
B A C D B
Participant 4
D A D C B
Participant 5
B C A D D
Enter Answer Key:
B C D A A
Scores:
Participant 1 = 0
Participant 2 = 1
Participant 3 = 1
Participant 4 = 1
Participant 5 = 2
Highest Score:
Participant 5
```

Program-12

Caesar Cipher is an encryption technique which is implemented as ROT13 ('rotate by 13 places'). It is a simple letter substitution cipher that replaces a letter with the letter 13 places after it in the alphabets, with the other characters remaining unchanged.

ROT13

A/a B/b C/c D/d E/e F/f G/g H/h I/i J/j K/k L/l
M/m

N/n O/o P/p Q/q R/r S/s T/t U/u V/v W/w X/x Y/y
Z/z

Write a program to accept a plain text of length L, where L must be greater than 3 and less than 100.

Encrypt the text if valid as per the Caesar Cipher.

Test your program with the sample data and some random data.

Example 1

INPUT:

Hello! How are you?

OUTPUT:

The cipher text is:

Uryyb! Ubj ner lbh?

Example 2

INPUT:

Encryption helps to secure data.

OUTPUT:

The cipher text is:

Rapelcgvba urycf gb frpher qngn.

Example 3

INPUT:

You

OUTPUT:
INVALID LENGTH

ALGORITHM

- Step-1 :- START
- Step-2 :- Create a class named as CaesarCipher.
- Step-3 :- Create a method named as main. In this function, create a variable named as text and assign a string value to it from the user (using Scanner class) which stores the plain text. Create a StringBuffer object named as sb. Create a for loop (from 0 to length of the string), inside this create a variable ch of type char and store each character of string text, now check the condition if the character ch is between A-M, if true then append((char)(ch+13)) to sb, else if the character ch is between N-Z, if true then append((char)(ch-13)) to sb, else append(ch) to sb. Now store the value of sb in a string variable named as cipher and print it.
- Step-4 :- END

VD TABLE

Sr. No.	Variable	Data Type	Description
1	text	String	Stores the plain text
2	sb	StringBuffer	Stores the cipher text
3	ch	char	Stores the character of the string
4	cipher	String	Stores the cipher text
5	i	int	Stores the index of the string
6	len	int	Stores the length of the string

OUTPUT

```
BlueJ: Terminal Window - basic
Options
Enter plain text:
Hello! How are you?
The cipher text is:
Uryyb! Ubj ner lbh?
```

Program-13

Design a class "Check" which checks whether a word is a palindrome or not.
(Palindrome words are those which spell the same from either ends).

Example: MADAM, LEVEL etc.

The details of the members of the class are given below:

Class name : Check

Data members / instance variables:

wrd : stores a word

len : to store the length of the word

Methods / Member functions:

Check() : default constructor

void acceptword() : to accept the word

boolean palindrome() : checks and returns true if the word is a palindrome otherwise returns false

void display() : displays the word along with an appropriate message

Specify the class "Check" giving details of the constructor, void acceptword(), boolean palindrome() and void display().

Define the main() function to create an object and call the functions accordingly to enable the task.

ALGORITHM

- Step-1 :- START
- Step-2 :- Create a class named as "Check".
- Step-3 :- Create a constructor to initialize the instance variable String wrd with null, int len with 0.
- Step-4 :- Create a void method "acceptword()" to input a word in wrd and and count the length of the word and store in len.
- Step-5 :- Create a boolean method "palindrome()" to check if the word is palindrome or not by checking the first and the last leter of the word i.e. palinedrome words will have first and last letter common.
- Step-6 :- Create a void method "display()" to print if the entered word is palindrome or not by checking the result of the boolean "palindrome()".
- Step-7 :- Create the "main" method to create a object and call "acceptword()" and "display()" methods.
- Step-8 :- END

VD TABLE

Sr. No.	Variable	Data Type	Description
1	wrd	String	Stores the input word
2	len	int	Stores the len of the input word
3	i	int	To iterate the for-loop for checking palindrome
4	j	int	To iterate the for-loop for checking palindrome

OUTPUT

```
BlueJ: Terminal Window - basic
Options
ENTER INTEGER TO CHECK (N): 197
197
971
719
197 IS A CIRCULAR PRIME.
```

Program-14

Design a class "Toggle" which toggles a word by converting all upper case alphabets to lower case and vice versa.

Example: The word mOTivATe becomes MotIVatE

The details of the members of the class are given below:

Class name : Toggle

Data members/instance variables:

str : stores a word

newstr : stores the toggled word

len : to store the length of the word

Methods/Member functions:

Toggle() : default constructor

void readword() : to accept the word

void toggle() : converts the upper case alphabets to lower case and all lower case alphabets to upper case and stores it in newstr

void display() : displays the original word along with the toggled word

Specify the class Toggle giving details of the constructor, void readword(), void toggle() and void display().

Define the main() function to create an object and call the functions accordingly to enable the task.

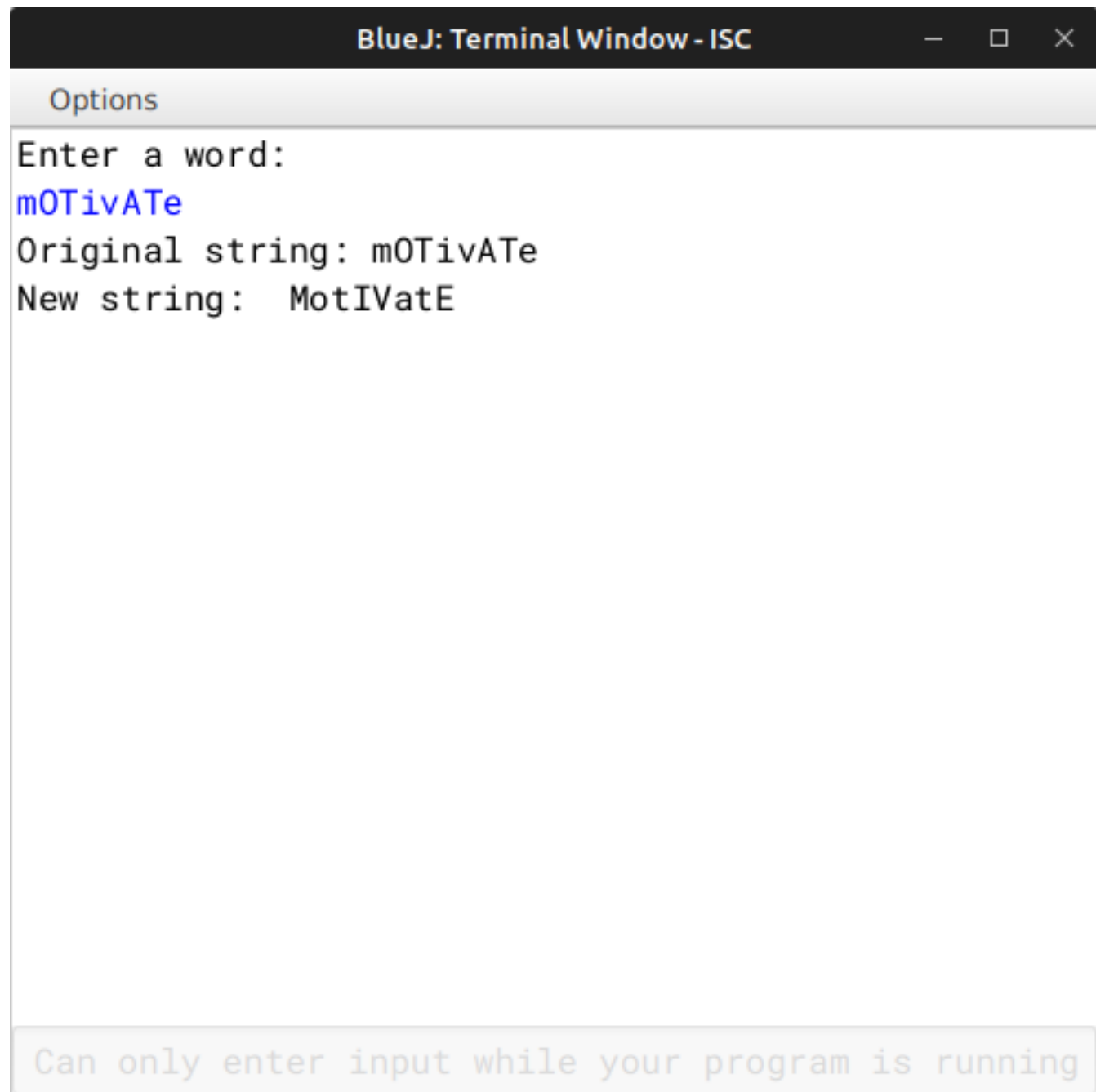
ALGORITHM

- Step-1 :- START
- Step-2 :- Create a class named as "Toggle".
- Step-3 :- Create a constructor to initialize the instance variable String str and newstr with null, int len with 0.
- Step-4 :- Create a void method "readword()" to input a word in str and count the length of the word and store in len.
- Step-5 :- Create a void method "toggle()" to go through the whole word and check if the letter is uppercase or lowercase and convert it to its opposite case.
- Step-6 :- Create a void method "display()" to print the original and the new word.
- Step-7 :- Create the "main" method to create a object and call "readword()", "toggle()" and "display()" methods.
- Step-8 :- END

VD TABLE

Sr. No.	Variable	Data Type	Description
1	str	String	Stores the input word
2	newstr	String	Stores the new toggled word
3	len	int	Stores the len of the input word
4	i	int	To iterate the for-loop for going through the word

OUTPUT



A screenshot of a BlueJ Terminal Window titled "BlueJ: Terminal Window - ISC". The window has a dark title bar with standard window controls (minimize, maximize, close). Below the title bar is a light gray bar labeled "Options". The main area of the terminal is white and contains the following text: "Enter a word:", "mOTivATe" (in blue), "Original string: mOTivATe", and "New string: MotIVatE". At the bottom of the terminal is a light gray bar with the text "Can only enter input while your program is running".

```
BlueJ: Terminal Window - ISC
```

Options

Enter a word:
mOTivATe
Original string: mOTivATe
New string: MotIVatE

Can only enter input while your program is running

Program-15

A class Fibo has been defined to generate the Fibonacci series 0, 1, 1, 2, 3, 5, 8, 13,.....
(Fibonacci series are those in which the sum of the previous two terms is equal to the next term).

Some of the members of the class are given below:

Class name : Fibo

Data member/instance variable:

start : integer to store the start value

end : integer to store the end value

Member functions/methods:

Fibo() : default constructor

void read() : to accept the numbers

int fibo(int n) : return the nth term of a Fibonacci series using recursive technique

void display() : displays the Fibonacci series from start to end by invoking the function

fibo()

Specify the class Fibo, giving details of the Constructor, void read(), int fibo(int), and void display().

Define the main() function to create an object and call the functions accordingly to enable the task.

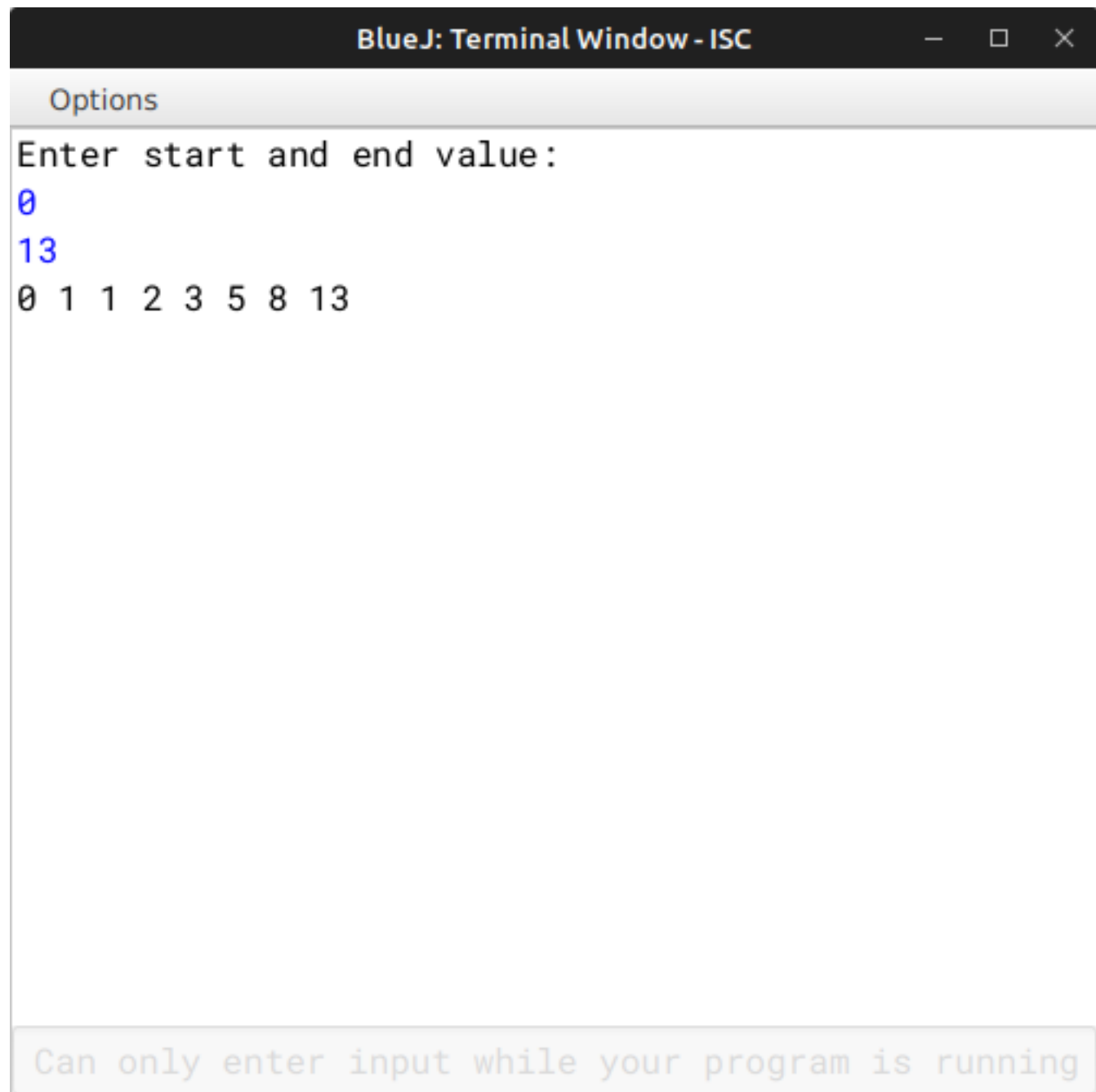
ALGORITHM

- Step-1 :- START
- Step-2 :- Create a class named as "Fibo".
- Step-3 :- Create a constructor to initialize the instance variable int start and end with 0.
- Step-4 :- Create a void method "read()" to input the start and end value for the series.
- Step-5 :- Create a int method "fibo(int n)" to return the nth term of a Fibonacci series using recursive technique.
- Step-6 :- Create a void method "display()" to displays the Fibonacci series from start to end by invoking the function fibo().
- Step-7 :- Create the "main" method to create a object and call "read" and "display" methods.
- Step-8 :- END

VD TABLE

Sr. No.	Variable	Data Type	Description
1	start	int	Store the start value
2	end	int	Store the end value
3	n	int	Formal parameter for method fibo()
4	i	int	To iterate the for-loop for printing the series int display()

OUTPUT



A screenshot of a BlueJ Terminal Window titled "BlueJ: Terminal Window - ISC". The window has a dark title bar with standard window controls (minimize, maximize, close). Below the title bar is a light gray bar labeled "Options". The main area of the terminal is white and contains the following text: "Enter start and end value:", followed by two lines of blue input text "0" and "13", and then a line of black output text "0 1 1 2 3 5 8 13". At the bottom of the terminal is a light gray bar with the text "Can only enter input while your program is running".

```
BlueJ: Terminal Window - ISC
```

Options

Enter start and end value:

0

13

0 1 1 2 3 5 8 13

Can only enter input while your program is running

Program-16

A class Gcd has been defined to find the Greatest Common Divisor of two integer numbers.

Some of the members of the class are given below:

Class name : Gcd

Data member/instance variable:

num1 : integer to store the first number

num2 : integer to store the second number

Member functions/methods:

Gcd() : default constructor

void accept() : to accept the numbers

int gcd(int x,int y) : return the GCD of the two number x and y using recursive technique

void display() : displays the result with an appropriate message

Specify the class Gcd, giving details of the Constructor, void accept(), int gcd(int,int), and void display().

Define the main() function to create an object and call the functions accordingly to enable the task.

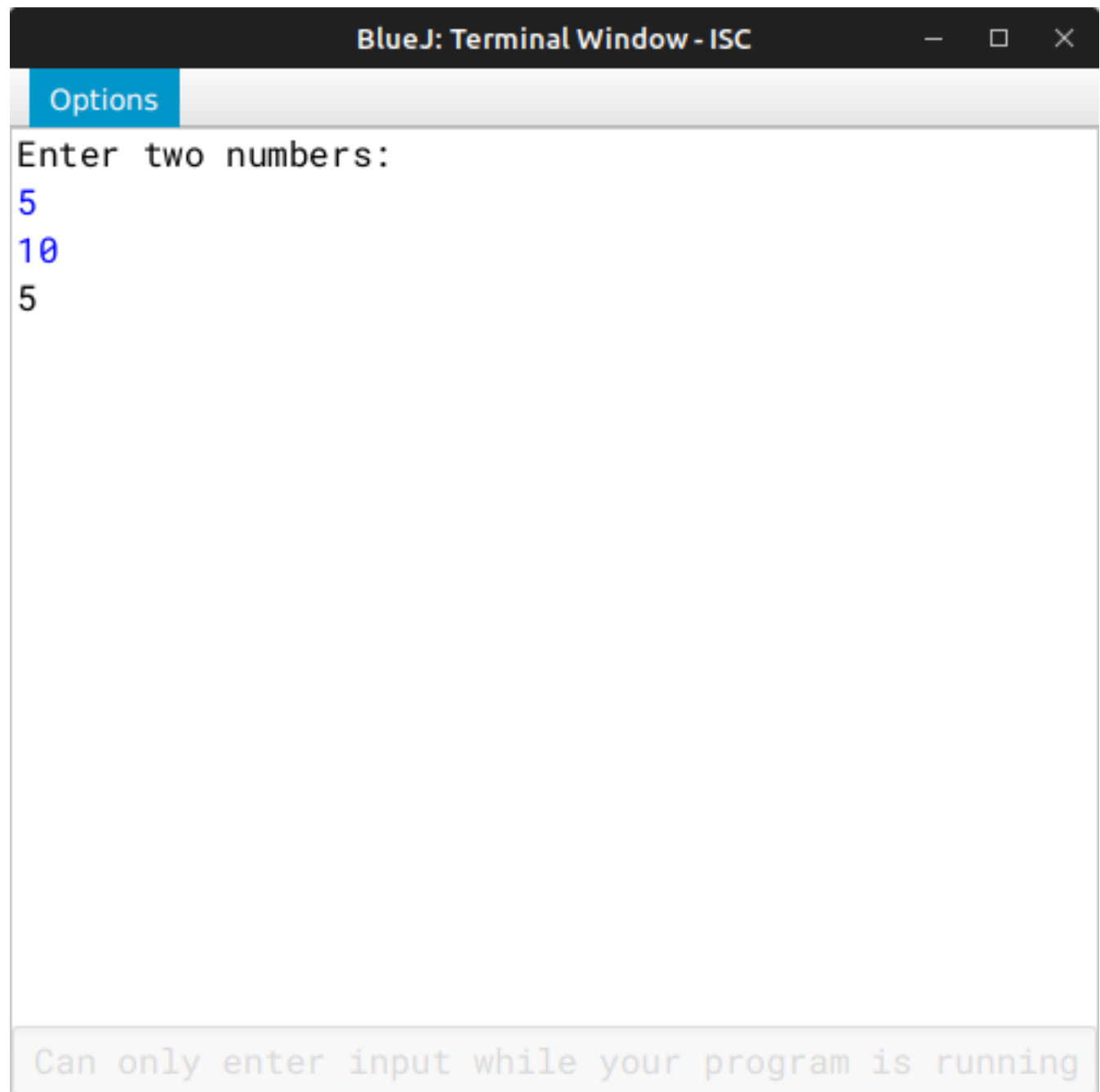
ALGORITHM

- Step-1 :- START
- Step-2 :- Create a class named as "Gcd".
- Step-3 :- Create a constructor to initialize the instance variable int num1 and num2 with 0.
- Step-4 :- Create a void method "accept" to accept two nos. in num1 and num2 respectively.
- Step-5 :- Create a int method "gcd" to calculate the gcd of the two nos. using recursive technique.
- Step-6 :- Create a void method "display" to print the gcd of the two nos.
- Step-7 :- Create the "main" method to create a object and call "accept" and "display" methods.
- Step-8 :- END

VD TABLE

Sr. No.	Variable	Data Type	Description
1	num1	int	Stores the first no.
2	num2	int	Stores the second no.
3	x	int	Formal parameter for function gcd()
4	y	int	Formal parameter for function gcd()
5	temp	int	Temporary variable for swapping num1 and num2 in display(), if num1 is smaller than num2

OUTPUT



A screenshot of a BlueJ Terminal Window titled "BlueJ: Terminal Window - ISC". The window has a dark title bar with standard window controls (minimize, maximize, close). Below the title bar is a light gray bar with a blue tab labeled "Options". The main area of the terminal is white and contains the text "Enter two numbers:" followed by three lines of input: "5", "10", and "5". At the bottom of the terminal is a light gray footer bar with the text "Can only enter input while your program is running".

```
BlueJ: Terminal Window - ISC
```

Options

Enter two numbers:

5

10

5

Can only enter input while your program is running

Program-17

Design a class ArmNum to check if a given number is an Armstrong number or not.
[A number is said to be Armstrong if sum of its digits raised to the power of length of the number is equal to the number]

Example : $371 = 3^3 + 7^3 + 1^3$
 $1634 = 1^4 + 6^4 + 3^4 + 4^4$
 $54748 = 5^5 + 4^5 + 7^5 + 4^5 + 8^5$

Thus 371, 1634 and 54748 are all examples of Armstrong numbers.

Some of the members of the class are given below:

Class name : ArmNum

Data members / instance variables:

n : to store the number

l : to store the length of the number

Methods / Member functions:

ArmNum(int nn) : parameterized constructor to initialize the data member n=nn

int sum_pow(int i) : returns the sum of each digit raised to the power of the length of the number using recursive technique eg. 34 will return $3^2 + 4^2$ (as the length of the number is 2)

void isArmstrong() : checks whether the given number is an Armstrong number by invoking the function sum_pow() and displays the result with an appropriate message

Specify the class ArmNum giving details of the constructor(), int sum_pow(int) and void isArmstrong().

Define a main() function to create an object and call the functions accordingly to enable the task.

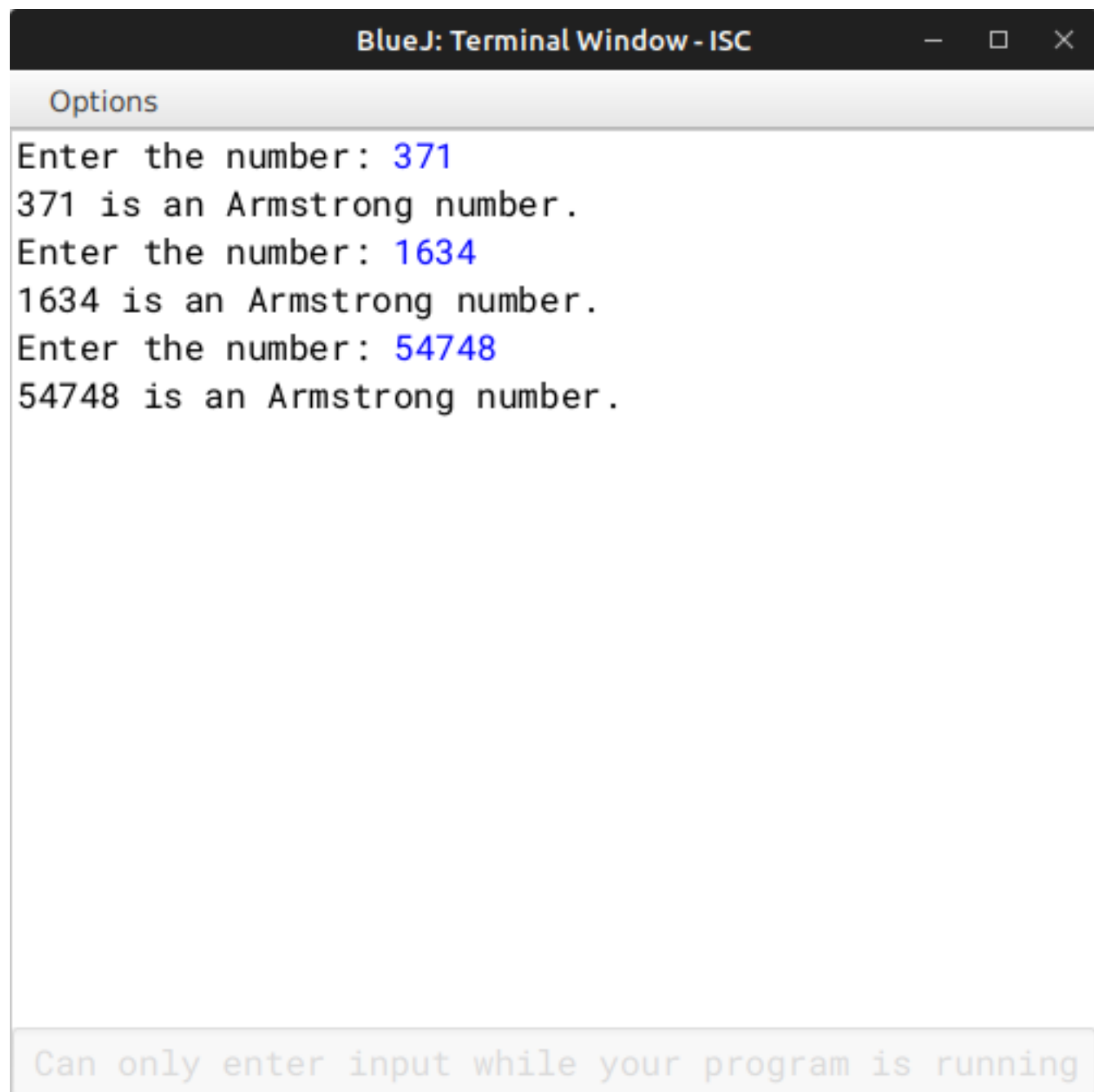
ALGORITHM

- Step-1 :- START
- Step-2 :- Create a class named as "ArmNum".
- Step-3 :- Create a parameterized constructor to initialize the instance variable int n with int num.
- Step-4 :- Create a int method "sum_pow(int i)" calculate the sum of the power of the digit to the length of digit, using recursive technique.
- Step-5 :- Create a void method "isArmstrong()" to check the no. is Armstrong and prints the appropriate message.
- Step-6 :- Create the "main" method to input the no. and create a object and call "isArmstrong()" methods.
- Step-7 :- END

VD TABLE

Sr. No.	Variable	Data Type	Description
1	n	int	Stores the no.
2	l	int	Stores the length of the no.
3	nn	int	Parameter for the parameterized constructor ArmNum(int nn)
4	c	int	Looping variable to count the length of the no.
5	i	int	Formal parameter for sum_pow(int i)

OUTPUT

A screenshot of a BlueJ terminal window titled "BlueJ: Terminal Window - ISC". The window has a dark title bar with standard window controls (minimize, maximize, close). Below the title bar is a light gray header labeled "Options". The main area of the window is white and contains the following text: "Enter the number: 371", "371 is an Armstrong number.", "Enter the number: 1634", "1634 is an Armstrong number.", and "Enter the number: 54748", "54748 is an Armstrong number.". The numbers 371, 1634, and 54748 are highlighted in blue. At the bottom of the window is a light gray footer with the text "Can only enter input while your program is running".

```
BlueJ: Terminal Window - ISC
```

Options

Enter the number: 371
371 is an Armstrong number.
Enter the number: 1634
1634 is an Armstrong number.
Enter the number: 54748
54748 is an Armstrong number.

Can only enter input while your program is running

Question 18:

Design a class **MatRev** to reverse each element of a matrix.

Example:

72	371	5	becomes	27	173	5
12	6	426		21	6	624
5	123	94		5	321	49

Some of the members of the class are given below:

Class name : **MatRev**

Data members/instance variables:

arr[][]	:	to store integer elements
m	:	to store the number of rows
n	:	to store the number of columns

Member functions/methods:

MatRev(int mm, int nn)	:	parameterised constructor to initialise the data
		members m = mm and n = nn
void fillarray()	:	to enter elements in the array
int reverse(int x)	:	returns the reverse of the number x
void revMat(MatRev P)	:	reverses each element of the array of the parameterized object and stores it in the array of the current object
void show()	:	displays the array elements in matrix form

Define the class **MatRev** giving details of the **constructor()**, **void fillarray()**, **int reverse(int)**, **void revMat(MatRev)** and **void show()**. Define the **main()** function to create objects and call the functions accordingly to enable the task.

ALGORITHM

- Step-1 :- START
- Step-2 :- Create a class named as "MatRev".
- Step-3 :- Create a parameterized constructor to initialize the instance variable int m and n and also initialize arr with m & n.
- Step-4 :- Create a void method "fillArray()" to accept elements in the array arr[m][n].
- Step-5 :- Create a int method "reverse(int x)" to reverse any no.
- Step-6 :- Create a void method "revMat(MatRev p)" to reverse the matrix with the help of "reverse(int x)".
- Step-7 :- Create a void method "show()" to display a matrix.
- Step-8 :- Create the "main" method to input the no. of rows and columns and create two objects obj1 & obj2 then take input in one array and fill the other array with the reverse integers of the first array, then print both.
- Step-9 :- END

VD TABLE

Sr. No.	Variable	Data Type	Description
1	arr[][]	int	A array to store integers in m rows and n columns
2	m	int	Stores no. of rows
3	n	int	Stores no. of columns
4	mm	int	Parameter for no. of rows in MatRev(int mm, int nn)
5	nn	int	Parameter for no. of columns in MatRev(int mm, int nn)
6	i	int	Looping variable in fillArray() & show()
7	j	int	Looping variable in fillArray() & show()
8	rev	int	Store the reverse of a no.
9	x	int	Store the user input no. of rows.
10	y	int	Store the user input no. of columns.

OUTPUT

```
BlueJ: Terminal Window - ISC
Options
Enter number of rows: 3
Enter number of columns: 3
Enter matrix elements:
72
371
5
12
6
426
5
123
94
Original Matrix is:-
72      371      5
12      6        426
5       123      94
Matrix with reversed elements:-
27      173      5
21      6        624
5       321      49
Can only enter input while your program is running
```

Program-19

A class Rearrange has been defined to modify a word by bringing all the vowels in the word at the beginning followed by the consonants.

Example: ORIGINAL becomes OIIARGNL

Some of the members of the class are given below:

Class name: Rearrange

Data member/instance variable:

wrđ: to store a word

newwrđ: to store the rearranged word

Member functions/methods:

Rearrange(): default constructor

void readword(): to accept the word in UPPER case

void freq_vow_con(): finds the frequency of vowels and consonants in the word and displays them with an appropriate message

void arrange(): the word by bringing the vowels at the beginning followed by consonants

void display(): the original word along with the rearranged word

Specify the class Rearrange, giving the details of the constructor(), void readword(), void freq_vow_con(), void arrange() and void display().

Define the main() function to create an object and call the functions accordingly to enable the task.

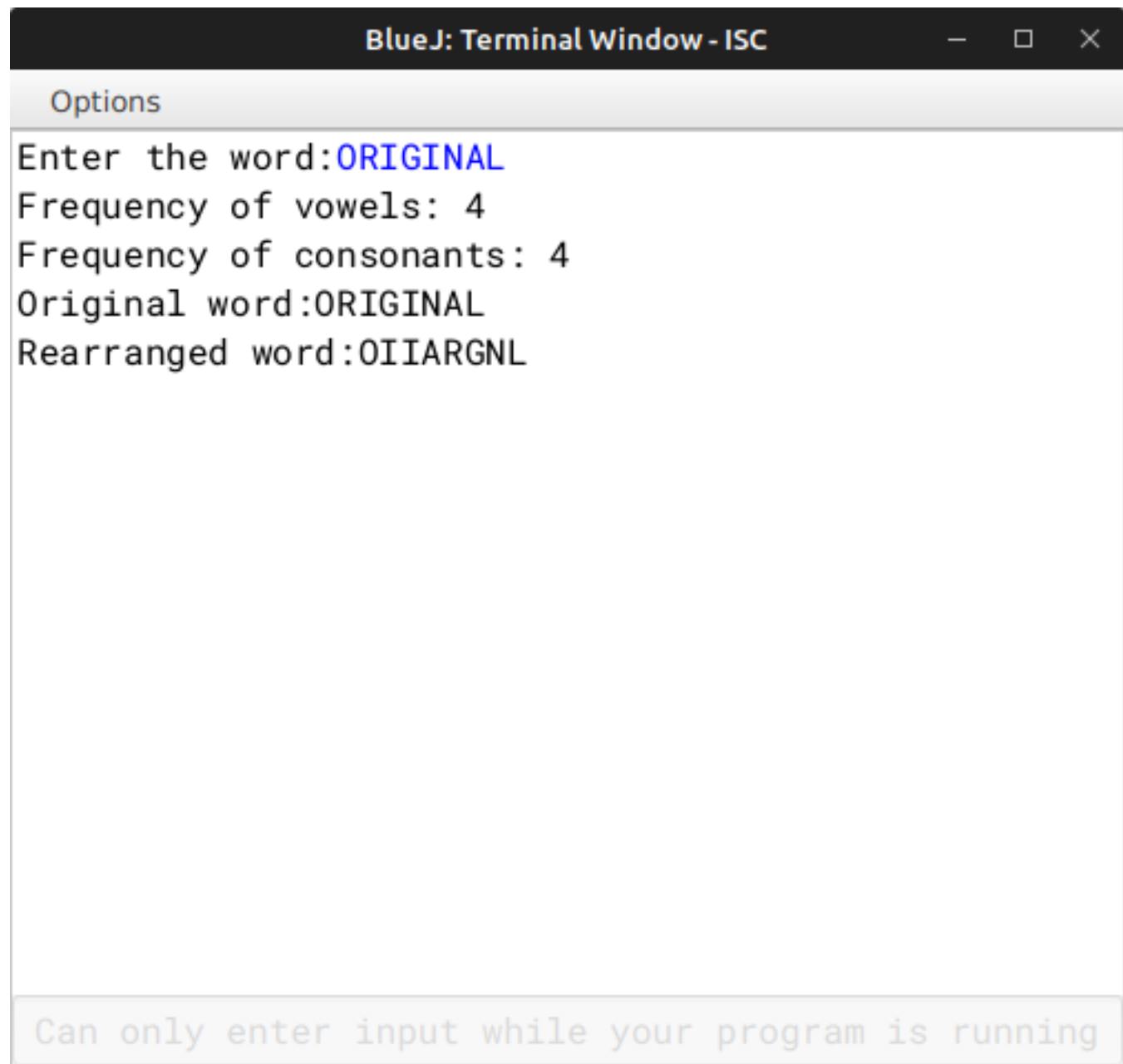
ALGORITHM

- Step-1 :- START
- Step-2 :- Create a class named as "Rearrange".
- Step-3 :- Create a constructor to initialize the instance variable String wrd and newwrd with null.
- Step-4 :- Create a void method "readword()" to accept the word and convert its case to Uppercase.
- Step-5 :- Create a void method "freq_vow_con()" to count the frequency of vowels and consonants.
- Step-6 :- Create a void method "arrange()" to arrange the letters by bringing the vowels at the beginning followed by consonants.
- Step-7 :- Create a void method "display()" to display the original and the rearranged word.
- Step-8 :- Create the "main" to make a object and call "readword()", "freq_vow_con()", "arrange()", and "display()".
- Step-9 :- END

VD TABLE

Sr. No.	Variable	Data Type	Description
1	wrd	String	Store the input word
2	newwrd	String	Store the rearranged word
3	v	int	Stores no. of vowels
4	c	int	Stores no. of consonants
5	v	String	Stores all the vowels of the word
6	c	String	Stores all the consonants of the word
7	i	int	Looping variable in freq_vow_con() & arrange()

OUTPUT

A screenshot of a BlueJ terminal window titled "BlueJ: Terminal Window - ISC". The window has a dark title bar with standard window controls (minimize, maximize, close). Below the title bar is a light gray header labeled "Options". The main area of the window is white and contains the following text: "Enter the word:ORIGINAL", "Frequency of vowels: 4", "Frequency of consonants: 4", "Original word:ORIGINAL", and "Rearranged word:OIIARGNL". At the bottom of the window is a light gray footer with the text "Can only enter input while your program is running".

```
BlueJ: Terminal Window - ISC
Options
Enter the word:ORIGINAL
Frequency of vowels: 4
Frequency of consonants: 4
Original word:ORIGINAL
Rearranged word:OIIARGNL
Can only enter input while your program is running
```


Program-20

Design a class Perfect to check if a given number is a perfect number or not.

[A number is said to be perfect if sum of the factors of the number excluding itself is equal to the original number]

Example: $6 = 1 + 2 + 3$ (where 1, 2 and 3 are factors of 6, excluding itself)

Some of the members of the class are given below:

Class name: Perfect

Data members/instance variables:

num: to store the number

Methods/Member functions:

Perfect (int nn): parameterized constructor to initialize the data member num=nn

int sum_of_factors(int i): returns the sum of the factors of the number(num), excluding itself, using a recursive technique

void check(): checks whether the given number is perfect by invoking the function sum_of_factors() and displays the result with an appropriate message

Specify the class Perfect giving details of the constructor(), int sum_of_factors(int) and void check().

Define a main() function to create an object and call the functions accordingly to enable the task.

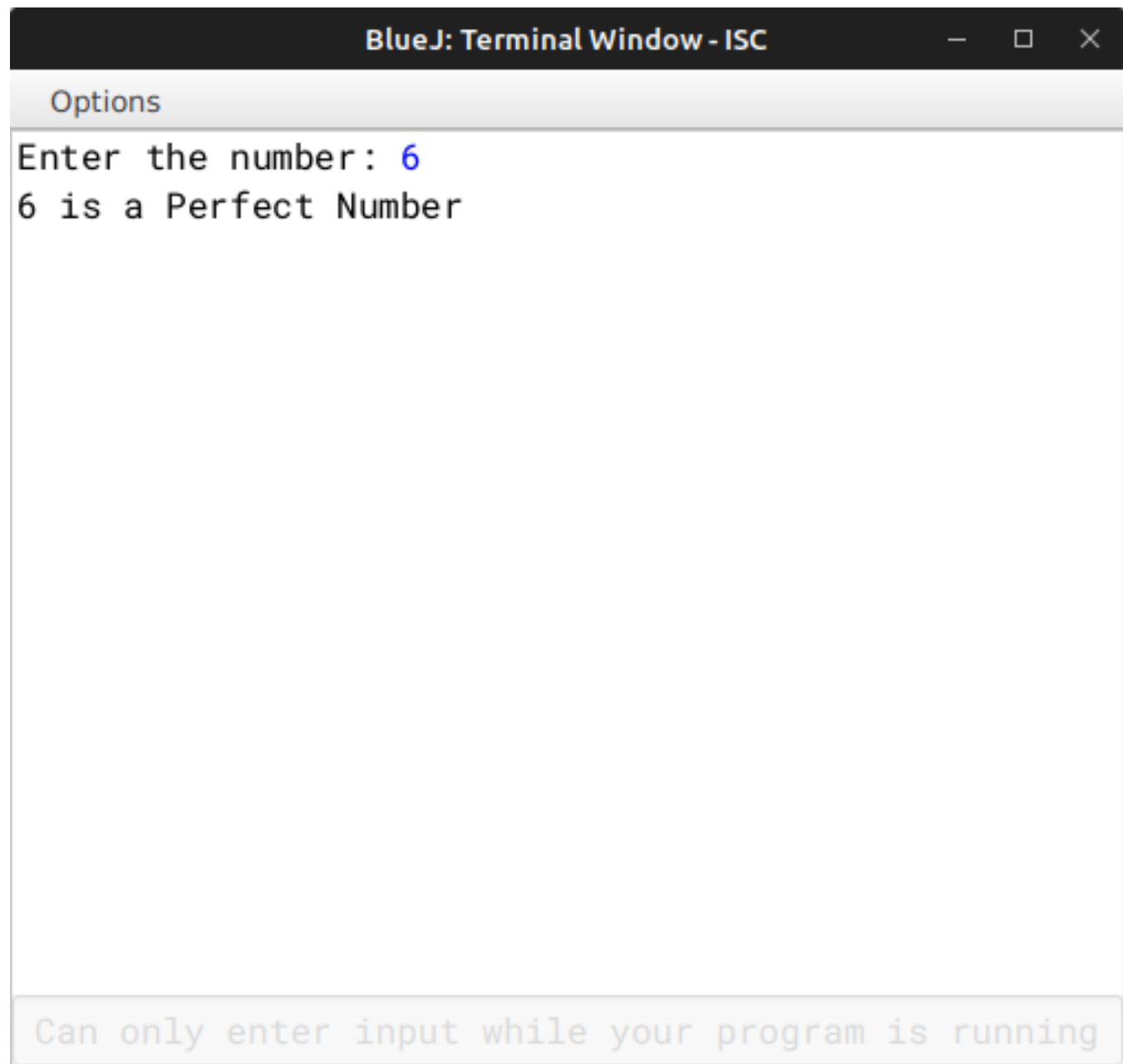
ALGORITHM

- Step-1 :- START
- Step-2 :- Create a class named as "Perfect".
- Step-3 :- Create a parameterized constructor to initialize the instance variable int num and f with nn and 1 respectively.
- Step-4 :- Create a int method "sum_of_factors(int i)" to sum up all the factors of a int i.
- Step-5 :- Create a void method "check()" to check the original no. is equal to the sum of factors of the no., and print the appropriate message.
- Step-6 :- Create the "main" to input the no. and pass it to the constructor and make a object and call the check() funtion.
- Step-7 :- END

VD TABLE

Sr. No.	Variable	Data Type	Description
1	num	int	Store the input no.
2	f	int	Increment variable for the no., to find all its factors
3	n	int	Stores the user input no. int the main method

OUTPUT



A screenshot of a BlueJ Terminal Window titled "BlueJ: Terminal Window - ISC". The window has a dark title bar with standard window controls (minimize, maximize, close). Below the title bar is a light gray bar labeled "Options". The main area of the terminal is white and contains the text "Enter the number: 6" followed by "6 is a Perfect Number" on the next line. The number "6" in the first line is blue. At the bottom of the terminal is a light gray bar with the text "Can only enter input while your program is running".

```
BlueJ: Terminal Window - ISC
```

Options

Enter the number: 6
6 is a Perfect Number

Can only enter input while your program is running

Question 21:

Two matrices are said to be equal if they have the same dimension and their corresponding elements are equal.

For example, the two matrices A and B is given below are equal:

Matrix A			Matrix B		
1	2	3	1	2	3
2	4	5	2	4	5
3	5	6	3	5	6

Design a class **EqMat** to check if two matrices are equal or not. Assume that the two matrices have the same dimension.

Some of the members of the class are given below :

Class name: **EqMat**

Data members/instance variables:

a[][] : to store integer elements

m: to store the number of rows

n: to store the number of columns

Member functions/methods:

EqMat(int mm, int nn): parameterized constructor to initialise the data members m = mm and n = nn

void readArray(): to enter elements in the array

int check(EqMat P, EqMat Q): checks if the parameterized objects P and Q are equal and returns 1 if true, otherwise returns 0

void print(): displays the array elements

Define the class **EqMat** giving details of the constructor **EqMat(int mm, int nn)**, **void readArray()**, **int check(EqMat, EqMat)** and **void print()**. Define the **main()** function to create objects and call the functions accordingly to enable the task.

ALGORITHM

- Step-1 :- START
- Step-2 :- Create a class named as "EqMat".
- Step-3 :- Create a parameterized constructor to initialize the instance variable int m, n and a[][] with mm, nn and a[][] with m and n.
- Step-4 :- Create a void method "readArray()" to input the elements of the arrays.
- Step-5 :- Create a boolean method "check(EqMat p, EqMat q)" to check if the elements of the array are equal or not.
- Step-6 :- Create a void method "print()" to print a matrix.
- Step-7 :- Create the "main" to user input the rows and columns for the matrixes and create two object for two matrixes and take input in those two matrix and print both matrixes and check if they are equal or not and print a appropriate message.
- Step-8 :- END

VD TABLE

Sr. No.	Variable	Data Type	Description
1	m	int	Store the no. of rows
2	n	int	Store the no. of columns
3	mm	int	Constructor parameter for no. of rows
4	nn	int	Constructor parameter for no. of columns
5	a[][]	int	Array to store the matrix of m rows and n columns
6	i	int	Looping variable in readArray(), check(EqMat p, EqMat q), and print()
7	j	int	Looping variable in readArray(), check(EqMat p, EqMat q), and print()
8	rows	int	Store the user input of rows
9	columns	int	Store the user input of columns

OUTPUT

```
BlueJ: Terminal Window - ISC
Options
Number of rows: 3
Number of columns: 3
Enter elements for first matrix:
1
2
3
4
5
6
7
8
9
Enter elements for second matrix:
1
2
3
4
5
6
7
8
9
First Matrix:
1      2      3
4      5      6
7      8      9
Second Matrix:
1      2      3
4      5      6
7      8      9
Both Matrices are Equal
Can only enter input while your program is running
```

Program-22

A class "Capital" has been defined to check whether a sentence has words beginning with a capital letter or not.

Some of the members of the class are given below:

Class name: Capital

Data member / instance variable:

sent: to store a sentence

freq: stores the frequency of words beginning with a capital letter

Member functions / methods:

Capital () : default constructor

void input () : to accept the sentence

boolean isCap(String w): checks and returns true if the word begins with a capital letter, otherwise returns false

void display(): displays the sentence along with the frequency of the words beginning with a capital letter

Specify the class Capital, giving the details of the constructor(), void input(), boolean isCap(String) and void display().

Define the main() function to create an object and call the functions accordingly to enable the task.

ALGORITHM

- Step-1 :- START
- Step-2 :- Create a class named as "Capital".
- Step-3 :- Create a parameterized constructor to initialize the instance variable String sent with null, and int freq with 0.
- Step-4 :- Create a void method "input()" to input the sentence.
- Step-5 :- Create a boolean method "isCap(String w)" to check if the first letter of the sentence is capital or not.
- Step-6 :- Create a void method "display()" to count the no. of words and check if the words start with a capital letter or not with the help of isCap(String w).
- Step-7 :- Create the "main" to create an object and call the "input()" & "output()" method.
- Step-8 :- END

VD TABLE

Sr. No.	Variable	Data Type	Description
1	sent	String	Store the input string
2	freq	int	Store the no. of words in the String
3	ct	int	Count the no. of words
4	w	String	Formal parameter for isCap(String w) method
5	i	int	Looping variable in display() for going through each word in the string and check them

OUTPUT

```
BlueJ: Terminal Window - ISC
Options
Enter one sentence
Your TIME is Limited, so don't waste it Living someone else's Life. - Steve Jobs
In "Your TIME is Limited, so don't waste it Living someone else's Life. - Steve Jobs".
The frequency of the words beginning with a capital letter is 7

Can only enter input while your program is running
```


Program-23

A Circular Prime is a prime number that remains prime under cyclic shifts of its digits. When the leftmost digit is removed and replaced at the end of the remaining string of digits, the generated number is still prime. The process is repeated until the original number is reached again.

A number is said to be prime if it has only two factors 1 and itself.

Example:

131

311

113

Hence, 131 is a circular prime.

Accept a positive number N and check whether it is a circular prime or not. The new numbers formed after the shifting of the digits should also be displayed.

Test your program with the following data and some random data:

Example 1

INPUT:

N = 197

OUTPUT:

197

971

719

197 IS A CIRCULAR PRIME.

Example 2

INPUT:

N = 1193

OUTPUT:

1193

1931

9311

3119

1193 IS A CIRCULAR PRIME.

Example 3

INPUT:

N = 29

OUTPUT:

29

92

29 IS NOT A CIRCULAR PRIME.

ALGORITHM

- Step-1 :- START
- Step-2 :- Create a class named as circular_prime.
- Step-3 :- Create a method named as is_prime and pass a parameter num. In this function, check whether the number is prime or not.
- Step-4 :- Create a method named as getDigitCount and pass a parameter num. The function returns the number of digits in the number.
- Step-5 :- Create a method named as main. In this function, take a number as input from the user using Scanner Class. Now check for the invalid input. If the input is invalid, then print the message Invalid Input. Now check whether the number is prime or not. If the number is prime, then check whether the number is circular prime or not. If the number is circular prime, then print the number in the rotated forms. If the number is prime even after rotating, then print the message Circular Prime. If the number is not prime, then print the message Not a Prime Number.
- Step-6 :- END

VD TABLE

Sr. No.	Variable	Data Type	Description
1	n	int	To store the number
2	num	int	To store the number
3	digitCount	int	To store the number of digits in the number
4	n2	int	To store the copy of the original number
5	divisor	int	To store the remainder
6	isCircularPrime	boolean	To store the boolean value
7	t1	int	Used as a temporary variable
8	t2	int	Used as a temporary variable
9	i	int	Used as loop variable

OUTPUT

```
BlueJ: Terminal Window - basic
Options
ENTER INTEGER TO CHECK (N): 131
131
311
113
131 IS A CIRCULAR PRIME.
```

Program-24

Write a program to declare a square matrix A[][] of order (M × M) where 'M' must be greater than 3 and less than 10. Allow the user to input positive integers into this matrix. Perform the following tasks on the matrix:

Sort the non-boundary elements in ascending order using any standard sorting technique and rearrange them in the matrix.

Calculate the sum of both the diagonals.

Display the original matrix, rearranged matrix and only the diagonal elements of the rearranged matrix with their sum.

Test your program for the following data and some random data:

Example 1

INPUT:

M = 4

```
9  2  1  5
8 13  8  4
15 6  3 11
7 12 23  8
```

OUTPUT:

ORIGINAL MATRIX

```
9  2  1  5
8 13  8  4
15 6  3 11
7 12 23  8
```

REARRANGED MATRIX

```
9  2  1  5
8  3  6  4
15 8 13 11
7 12 23  8
```

DIAGONAL ELEMENTS

```
9      5
  3  6
  8 13
7      8
```

SUM OF THE DIAGONAL ELEMENTS = 59

Example 2

INPUT:

M = 5

```
7 4 1 9 5
8 2 6 10 19
13 1 3 5 1
10 0 5 12 16
1 8 17 6 8
```

OUTPUT:

ORIGINAL MATRIX

```
7 4 1 9 5
8 2 6 10 19
13 1 3 5 1
10 0 5 12 16
1 8 17 6 8
```

REARRANGED MATRIX

```
7 4 1 9 5
8 0 1 2 19
13 3 5 5 1
10 6 10 12 16
1 8 17 6 8
```

DIAGONAL ELEMENTS

```
7      5
 0    2
    5
 6   12
1      8
```

SUM OF THE DIAGONAL ELEMENTS = 46

Example 3

INPUT:

M = 3

OUTPUT:

THE MATRIX SIZE IS OUT OF RANGE.

ALGORITHM

- Step-1 :- START
- Step-2 :- Create a class named as rearrange.
- Step-3 :- Create a method named as sortNonBoundaryMatrix passing an array and an integer as parameters. In this function, sort the non-boundary elements of the matrix in ascending order by first converting the 2D array into 1D array and then sorting it and then again transferring it to 2D array.
- Step-4 :- Create a method named as computePrintDiagonalSum passing an array and an integer as parameters. In this function, compute the sum of the diagonal elements of the matrix and print the final sum.
- Step-5 :- Create a method named as printMatrix passing an array and an integer as parameters. In this function, print the matrix.
- Step-6 :- Create a method named as main. In this function, input the size of the matrix and the elements of the matrix from the user using the Scanner class. Then, call the sortNonBoundaryMatrix function and then the computePrintDiagonalSum function and finally call the printMatrix function.
- Step-7 :- END

VD TABLE

Sr. No.	Variable	Data Type	Description
1	m	int	To store the size of the matrix
2	a	int[][]	Array to store the elements of the matrix
3	i	int	Loop variable
4	j	int	Loop variable
5	t	int	Temporary variable to store the elements of the matrix
6	sum	int	To store the sum of the diagonal elements of the matrix
7	k	int	Temporary Variable
8	b	int[]	Array to store the elements of the matrix (1D Format)

OUTPUT

```
BlueJ: Terminal Window - basic
Options
ENTER MATRIX SIZE (M): 4
ENTER ELEMENTS OF MATRIX
ENTER ROW 1:
9 2 1 5
ENTER ROW 2:
8 13 8 4
ENTER ROW 3:
15 6 3 11
ENTER ROW 4:
7 12 23 8
ORIGINAL MATRIX
9      2      1      5
8      13     8      4
15     6      3     11
7      12     23     8
REARRANGED MATRIX
9      2      1      5
8      3      6      4
15     8      13     11
7      12     23     8
DIAGONAL ELEMENTS
9              5
      3      6
      8      13
7              8
SUM OF THE DIAGONAL ELEMENTS = 59
```

Program-25

Write a program to accept a sentence which may be terminated by either '.', '?' or '!' only. The words may be separated by more than one blank space and are in UPPER CASE.

Perform the following tasks:

Find the number of words beginning and ending with a vowel.

Place the words which begin and end with a vowel at the beginning, followed by the remaining words as they occur in the sentence.

Test your program with the sample data and some random data:

Example 1

INPUT:

ANAMIKA AND SUSAN ARE NEVER GOING TO QUARREL ANYMORE.

OUTPUT:

NUMBER OF WORDS BEGINNING AND ENDING WITH A VOWEL = 3

ANAMIKA ARE ANYMORE AND SUSAN NEVER GOING TO QUARREL

Example 2

INPUT:

YOU MUST AIM TO BE A BETTER PERSON TOMORROW THAN YOU ARE TODAY.

OUTPUT:

NUMBER OF WORDS BEGINNING AND ENDING WITH A VOWEL = 2

A ARE YOU MUST AIM TO BE BETTER PERSON TOMORROW THAN YOU TODAY

Example 3

INPUT:

LOOK BEFORE YOU LEAP.

OUTPUT:

NUMBER OF WORDS BEGINNING AND ENDING WITH A VOWEL = 0

LOOK BEFORE YOU LEAP

Example 4

INPUT:
HOW ARE YOU@

OUTPUT:
INVALID INPUT

ALGORITHM

- Step-1 :- START
- Step-2 :- Create a class named as vowel.
- Step-3 :- Create a method named as isVowel and pass a character as a parameter. In this function, check if the character is a vowel or not.
- Step-4 :- Create a method named as main. In this function, input the sentence from the user using Scanner class. Now check whether the sentence is valid or not. Now place the words which begin and end with a vowel at the beginning, followed by the remaining words as they occur in the sentence. Print the sentence and number of words beginning and ending with a vowel.
- Step-5 :- END

VD TABLE

Sr. No.	Variable	Data Type	Description
1	ipStr	String	To store the sentence
2	word	String	To store the words
3	str	char	To store the trimmed sentence
4	c	int	Used as counter variable
5	wordLen	int	To store the length of each word
6	newStr	String	To store the changed sentence
7	len	int	To store the length of the sentence

OUTPUT

```
BlueJ: Terminal Window - basic
Options
ENTER THE SENTENCE:
ANAMIKA AND SUSAN ARE NEVER GOING TO QUARREL ANYMORE.
NUMBER OF WORDS BEGINNING AND ENDING WITH A VOWEL = 3
ANAMIKA ARE ANYMORE AND SUSAN NEVER GOING TO QUARREL
```

Program-26

A class Adder has been defined to add any two accepted time.

Example: Time A 6 hours 35 minutes

Time B 7 hours 45 minutes

Their sum is 14 hours 20 minutes (where 60 minutes = 1 hour)

The details of the members of the class are given below:

Class name Adder

Data member/instance variable:

a[] integer array to hold two elements (hours and minutes)

Member

functions/methods:

Adder() constructor to assign 0 to the array elements

void readtime() to enter the elements of the array

void addtime(Adder X, Adder Y) adds the time of the two parameterized objects X and Y and stores the sum in the current calling object

void disptime() displays the array elements with an appropriate message (i.e. hours = and minutes =)

Specify the class Adder giving details of the constructor(), void readtime(), void addtime(Adder, Adder) and void disptime(). Define the main() function to create objects and call the functions accordingly to enable the task.

ALGORITHM

- Step-1 :- START
- Step-2 :- Create a class named as Adder.
- Step-3 :- Declare an array of type interger.
- Step-4 :- Create a constructor named as adder. In this constructor, initialize the array with size 2.
- Step-5 :- Create a method named as readtime. In this function, take the time input from the user using Scanner class.
- Step-6 :- Create a method named as add passing two parameterised objects X and Y. In this function, adds the time of the two parameterized objects X and Y and stores the sum in the current calling object.
- Step-7 :- Create a method named as display. In this function, display the time.
- Step-8 :- Create a method named as main. In this function, create two objects of the class Adder and call the methods.
- Step-9 :- END

VD TABLE

Sr. No.	Variable	Data Type	Description
1	a	int[]	To store the time
2	a	int	To store the hour
3	b	int	To store the minute
4	c	int	To store the addition of time of the two objects

OUTPUT

```
BlueJ: Terminal Window - basic
Options
1st Time Enter Hours
6
Enter Minutes
35
2nd Time Enter Hours
7
Enter Minutes
45

1st Time
6 Hours 35 Minutes
2nd Time
7 Hours 45 Minutes
Added Time
14 Hours 20 Minutes
```

Program-27

A class SwapSort has been defined to perform string related operations on a word input.

Some of the members of the class are as follows:

Class name SwapSort

Data members/instance

variables:

wrd to store a word

len integer to store length of the word

swapwrd to store the swapped word

sortwrd to store the sorted word

Member functions/methods:

SwapSort() default constructor to initialize data members with legal initial values

void readword() to accept a word in UPPER CASE

void swapchar() to interchange/swap the first and last characters of the word in wrd
and stores the new word in swapwrd

void sortword() sorts the characters of the original word in alphabetical order and
stores it in sortwrd

void display() displays the original word, swapped word and the sorted word

Specify the class SwapSort, giving the details of the constructor(), void readword(), void swapchar(), void sortword() and void display(). Define the main() function to create an object and call the functions accordingly to enable the task.

ALGORITHM

- Step-1 :- START
- Step-2 :- Create a class named as SwapSort.
- Step-3 :- Declare wrd to store a word, len integer to store length of the word, swapwrd to store the swapped word, sortwrd to store the sorted word.
- Step-4 :- Create a constructor named as SwapSort to initialize data members with legal initial values.
- Step-5 :- Create a method named as readword to accept a word in UPPER CASE.
- Step-6 :- Create a method named as swapchar to interchange/swap the first and last characters of the word in wrd and stores the new word in swapwrd.
- Step-7 :- Create a method named as sortchar to sort the characters of the word in swapwrd and stores the new word in sortwrd.
- Step-8 :- Create a method named as main to create an object and call the functions accordingly to enable the task.
- Step-9 :- END

VD TABLE

Sr. No.	Variable	Data Type	Description
1	wrd	String	To store a word
2	len	int	To store length of the word
3	swapwrd	String	To store the swapped word
4	sortwrd	String	To store the sorted word
5	i	int	To store the index of the word
6	j	int	To store the index of the word
7	c	char	To store the temporary character

OUTPUT

```
BlueJ: Terminal Window - basic
Options
Enter word in Upper case
HERO
Original word = HERO
Swapped word = OERH
Sorted word = EHOR
```

Program-28

A disarium number is a number in which the sum of the digits to the power of their respective position is equal to the number itself.

Example: $135 = 1^1 + 3^2 + 5^3$

Hence, 135 is a disarium number.

Design a class Disarium to check if a given number is a disarium number or not. Some of the members of the class are given below:

Class name : Disarium

Data members/instance variables:

int num : stores the number

int size : stores the size of the number

Methods/Member functions:

Disarium(int nn) : parameterized constructor to initialize the data members n = nn and size = 0

void countDigit() : counts the total number of digits and assigns it to size

int sumofDigits(int n, int p) : returns the sum of the digits of the number(n)

void check() : checks whether the number is a disarium number and displays the result with an appropriate message

Specify the class Disarium giving the details of the constructor(), void countDigit(), int sumofDigits(int, int) and void check(). Define the main() function to create an object and call the functions accordingly to enable the task.

ALGORITHM

- Step-1 :- START
- Step-2 :- Create a class named as Disarium.
- Step-3 :- Declare variables - num to store the number and size to store the size of the number.
- Step-4 :- Create a constructor named as Disarium to initialize the variables to initial values.
- Step-5 :- Create a method named as countDigits to count the number of digits in the number.
- Step-6 :- Create a method named as sumofDigits with two interger type parameters to return the sum of the digits of the number.
- Step-7 :- Create a method named as check to check whether the number is a disarium number and display the result with an appropriate message.
- Step-8 :- Create a method named as main to to create an object and call the functions accordingly to enable the task.
- Step-9 :- END

VD TABLE

Sr. No.	Variable	Data Type	Description
1	num	int	To store the number
2	size	int	To store the size of the number
3	m	int	To store the input number by the user
4	a	int	Temporary variable

OUTPUT

```
BlueJ: Terminal Window - basic
Options
Input a Number
89
Disarium Number
```


Program-29

A class Shift contains a two dimensional integer array of order (m n) where the maximum values of both m and n is 5. Design the class Shift to shuffle the matrix (i.e. the first row becomes the last, the second row becomes the first and so on). The details of the members of the class are given below:

Class name : Shift

Data member/instance variable:

mat[][] : stores the array element

m : integer to store the number of rows

n : integer to store the number of columns

Member functions/methods:

Shift(int mm, int nn) : parameterized constructor to initialize the data

members m = mm and n = nn

void input() : enters the elements of the array

void cyclic(Shift P) : enables the matrix of the object(P) to shift each row upwards in a cyclic manner and store the resultant matrix in the current object

void display() : displays the matrix elements

Specify the class Shift giving details of the constructor(), void input(), void cyclic(Shift) and void display(). Define the main() function to create an object and call the methods accordingly to enable the task of shifting the array elements.

ALGORITHM

- Step-1 :- START
- Step-2 :- Create a class named as Shift.
- Step-3 :- Declare variables - mat[][] to stores the array elements, m to store the number of rows and n to store the number of columns.
- Step-4 :- Create a constructor named as Shift with two integer type intergers, to initialize the variables to initialize the data.
- Step-5 :- Create a method named as input to input the elements of the array.
- Step-6 :- Create a method named as cyclic to enable the matrix of the object(P) to shift each row upwards in a cyclic manner and store the resultant matrix in the current object.
- Step-7 :- Create a method named as display to display the elements of the array.
- Step-8 :- Create a method named as main to create an object of the class Shift and call the methods.
- Step-9 :- END

VD TABLE

Sr. No.	Variable	Data Type	Description
1	mat[][]	int	To store the array elements
2	m	int	To store the number of rows
3	n	int	To store the number of columns
4	i	int	To store the row number
5	j	int	To store the column number
6	mm	int	To store the number of rows inside the constructor
7	nn	int	To store the number of columns inside the constructor - Shift

OUTPUT

```
BlueJ: Terminal Window - basic
Options
Enter elements
1 2 3 4
5 6 7 8
9 10 11 12
Output:
1      2      3      4
5      6      7      8
9      10     11     12
Output:
5      6      7      8
9      10     11     12
1      2      3      4
```

Program-30

A class ConsChange has been defined with the following details:

Class name : ConsChange

Data members/instance variables:

word : stores the word

len : stores the length of the word

Member functions/methods:

ConsChange() : default constructor

void readword() : accepts the word in lowercase

void shiftcons() : shifts all the consonants of the word at the beginning followed by the vowels (e.g. spoon becomes spnoo)

void changeword() : changes the case of all occurring consonants of the shifted word to uppercase, for e.g. (spnoo becomes SPNoo)

void show() : displays the original word, shifted word and the changed word

Specify the class ConsChange giving the details of the constructor(),

void readword(), void shiftcons(), void changeword() and void show(). Define the main() function to create an object and call the functions accordingly to enable the task.

ALGORITHM

- Step-1 :- START
- Step-2 :- Create a class named as ConsCharge.
- Step-3 :- Declare variables - word to store the word and len to store the length of the word.
- Step-4 :- Create a constructor named as ConsCharge to initialize the variables to default values.
- Step-5 :- Create a method named as readword to accept the word in lowercase.
- Step-6 :- Create a method named as shiftcons to shift all the consonants of the word at the beginning followed by the vowels.
- Step-7 :- Create a method named as changeword to change the case of all occurring consonants of the shifted word to uppercase.
- Step-8 :- Create a method named as show to display the original word, shifted word and the changed word.

VD TABLE

Sr. No.	Variable	Data Type	Description
1	word	String	To store the word
2	len	int	To store the length of the word
3	i	int	To store the index of the word
4	c	char	To store the character of the word
5	s	String	To store the shifted word (in shiftcons() function)
6	s	String	To store the shifted word (in changeword() function)

OUTPUT

```
BlueJ: Terminal Window - basic
Options
Enter word in Lower case
spoon

Original word= spoon
Sorted Word=spnoo
Changed word= SPNoo
```