

THE MODERN ACADEMY

COMPUTER SCIENCE PROJECT



NAME: Saptarshi Chattopadhyay

CLASS: XII

SEC: B(Sc)

ROLL No. : 20

INDEX

Serial Number	Question	Page Number
1	Emirp Number	3
2	Circular Prime Number	5
3	Filling a matrix with three unique characters	9
4	Sum of row and column	12
5	Counting consonants in a String	16
6	Happy number	19
7	Counting frequency of "an" and "and"	22
8	Checking if two matrices are equal	25
9	Adding two times	28
10	Finding max, min of individual rows and columns	31
11	Function Overloading	34
12	Bubble, Insertion, Selection Sorting	36
13	Convert Binary number to Decimal and vice versa	40
14	Disarium number (using recursion)	42
15	Recursive Binary Search	43
16	Sum, Difference and product of two matrices	47
17	Valid Date checking	52
18	Program to find area of a circle using coordinates of two points as diameter	55
19	Circular shift of a matrix	59
20	Series Program	63
21	ISBN number	66
22	Inheritance Program(Series)	70
23	Wordpile	73
24	Queue	77
25	String program to implement queue	81
26	Program to find the least no. of cartons required for N no. of boxes	85
27	Quiz Competition	88
28	Caesar encryption (ROT13)	93

Q1) An emirp number is a number which is prime backwards and forwards.
Example : 13 and 31 are both prime numbers. Thus 13 is a emirp number.
Write a java program to check whether a number is emirp number or not.

Example:

13, 17, 31, 37, 71, 73, 79, 97, 107, 113, 149, 157, 167, 179, 199

Algorithm

1. START
2. Take input form user
3. Execute a loop from $i=2$ to $i<(n/2)$
4. Check if n is divisible by i from 2 to $n/2$
5. If i is not divisible by any value of i then n is prime
6. If i is divisible by any value of i then print "invalid input" and exit program
7. To find the reverse of n calculate the $n\%10$ and add to the reverse number
8. Update n by $n/10$
9. Now check if the reverse number is prime or not
10. Repeat steps 3 to 5 to check for prime number
11. If reverse number is prime number print n is an Emirp number otherwise print n is not an Emirp number
12. END

Source code

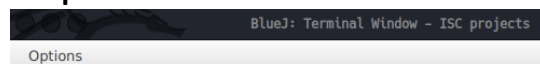
```
import java.util.Scanner;
public class Emirp{
public static boolean isPrime(int num){
if(num<=1)
return false;
for(int i=2;i<(num/2);i++){
if(num%i==0)
return false;}
return true;}
public static void main(String args[]){
Scanner nrt=new Scanner(System.in);
System.out.println("enter a number");
int user=nrt.nextInt();
if(isPrime(user)){
//reversing the number
int copy=user,reverse=0;
while(copy!=0){
reverse = (reverse*10) +(copy%10);
copy/=10;}
if(isPrime(reverse))
System.out.println(user+" is an Emirp Number");
else
```

```
System.out.println(user+" is NOT an Emirp Number");}
else
System.out.println("Invalid Input");}}
```

Variable Description Table

Variable name	Data type	Use
num	int	Argument for isPrime method
nrt	Wrapper object	For taking input
user	int	To store the input
copy	int	To store a copy of user variable's value
reverse	int	To store the reverse of user

Output



```
Options
enter a number
13
13 is an Emirp Number
enter a number
17
17 is an Emirp Number
enter a number
31
31 is an Emirp Number
enter a number
37
37 is an Emirp Number
enter a number
71
71 is an Emirp Number
enter a number
73
73 is an Emirp Number
enter a number
79
79 is an Emirp Number
enter a number
97
97 is an Emirp Number
enter a number
107
107 is an Emirp Number
```

```
enter a number
113
113 is an Emirp Number
enter a number
149
149 is an Emirp Number
enter a number
157
157 is an Emirp Number
enter a number
167
167 is an Emirp Number
enter a number
179
179 is an Emirp Number
enter a number
199
199 is an Emirp Number
```

Q2) 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. Write a program to 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

1. Start.
2. Input a number n.
3. Find if the number is prime.
4. Declare a function isPrime() to check prime.
5. Execute loop from i=1 to i<=n.
6. If n is divisible by any i then count the number of times it gets divisible by doing c++.
7. If c==2, then the number n is prime .
8. If not then stop otherwise go to next step.
9. Calculate the length (l) of the number by converting n into string.
10. Then find the divisor=(int)(Math.pow(10,l-1)).
11. Store a copy of n in m.

12. Execute a loop from $i=0$ to $i<l$ to generate new circulated numbers.
13. Calculate the quotient= $n1/\text{divisor}$ and the remainder= $n2\%\text{divisor}$.
14. To generate the new circulated number(m) follow the next step.
15. $m=r*10+n1$;
16. Call the function `isPrime()` to check whether new number(m) is prime.
17. If m is not prime then break;
18. If all the numbers are prime then print it is a circular prime otherwise not a circular prime.
19. Stop

Source Code

```
import java.util.Scanner;
class CircularPrime{
    static boolean isPrime(int num) {
        int c = 0;
        for (int i = 2; i <= num/2; i++)
        { if (num % i == 0){
            c++;
        }}
        if(c==2)
        System.out.println(num+"is prime");
        else
        System.out.println(num+"is not prime");
        return c == 2;
    }
    public static void main(String args[]){
        Scanner in = new Scanner(System.in);
        System.out.print("Enter the number: ");
        int n = in.nextInt();
        int f=1;
        if (isPrime(n)){
            //System.out.println(n);
            String s=Integer.toString(n);
            int l=s.length();
            int divisor = (int)(Math.pow(10, l- 1));
            int m = n;
            for (int i = 1; i < l; i++){
                int n1 = m / divisor;
                int n2 = m % divisor;
                m = n2 * 10 + n1;
                //System.out.println(m);
                if (!isPrime(m)) {
                    f=0;
                    break;}}}
        else { f=0;}
```

```

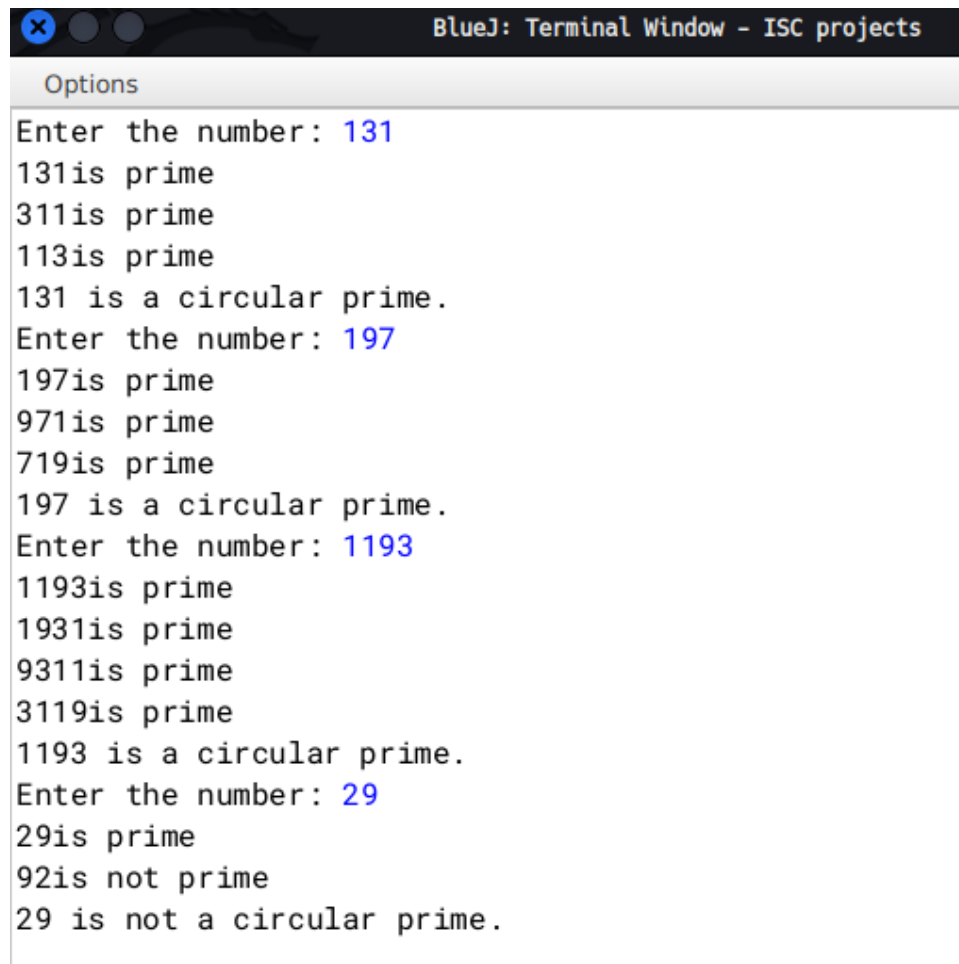
if (f==1){
    System.out.println(n + " is a circular prime.");}
else {
    System.out.println(n + " is not a circular prime."); }}

```

Variable Description Table

Variable name	Data type	Use
num	int	Argument for isPrime method
in	Wrapper object	For taking input
n	int	To store the input
m	int	Stores a copy of n
n1	int	Stores quotient
n2	int	divisor
copy	int	To store a copy of user variable's value
c	int	To check if a number is prime or not
l	int	Stores length of n
s	String	Stores String form of n
divisor	int	Stores the divisor

Output



```
Options
Enter the number: 131
131is prime
311is prime
113is prime
131 is a circular prime.
Enter the number: 197
197is prime
971is prime
719is prime
197 is a circular prime.
Enter the number: 1193
1193is prime
1931is prime
9311is prime
3119is prime
1193 is a circular prime.
Enter the number: 29
29is prime
92is not prime
29 is not a circular prime.
```


Q3) Write a program to declare a square matrix M [] [] of order 'N' where 'N' must be greater than 3 and less than 10. Allow the user to accept three different characters from the keyboard and fill the array according to the instruction given below:

(i) Fill the four corners of the square matrix by character 1.

(ii) Fill the boundary elements of the matrix (except the four corners) by character 2.

(iii) Fill the non-boundary elements of the matrix by character 3.

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

INPUT: N = 5

FIRST CHARACTER: A

SECOND CHARACTER: C

THIRD CHARACTER: X

OUTPUT:

A C C C A

C X X X C

C X X X C

C X X X C

A C C C A

INPUT: N = 4

FIRST CHARACTER: @

SECOND CHARACTER: ?

THIRD CHARACTER: #

OUTPUT:

@ ? ? @

? # # ?

? # # ?

@ ? ? @

Algorithm

1 START

2 Ask user to input three characters.

3 Ask user to input the order of matrix 'M'.

4 Declare a two dimensional array of type char with name 'M'.

5 Start a for loop for(int i=0;i<n;i++).

6 Start a for loop inside as for(int j=0;j<n;j++)

7 If (i and j are both zero) or (either i is zero and j is n-1) or (i is n-1 and j is zero) or (both i and j are n-1), then

fill two dimensional array M at location M[i][j] with the first character the user has given as input.

8 If i and j lies between 1 and n-2 then fill two dimensional array M at location M[i][j] with the second character user has given.
 9 Else fill M[i][j] with the third character given as input by the user.
 10 Run two nested loops as for(int i=0;i<n;i++) and inside for(int j=0;j<n;j++) and print all elements.
 11 END

Source Code

```
import java.util.Scanner;
public class Matrix{
    public static void main(){
        Scanner nrt=new Scanner(System.in);
        System.out.println("Enter three characters");
        char ch1,ch2,ch3;
        ch1=(nrt.nextLine()).charAt(0);
        ch2=(nrt.nextLine()).charAt(0);
        ch3=(nrt.nextLine()).charAt(0);
        System.out.println("Enter order of Matrix");
        int n=nrt.nextInt();
        char M[][]= new char[n][n];
        for(int i=0;i<n;i++){
            for(int j=0;j<n;j++){
                if( (i==0 && j==0) || (i==0 && j==n-1) || (i==n-1 && j==0) || (i==n-1 && j==n-1))
                    M[i][j]=ch1;
                else if ( (i>=1 && i<=(n-2)) && (j>=1 && j<=(n-2)))
                    M[i][j]=ch2;
                else
                    M[i][j]=ch3;}}
        for(int i=0 ;i<n;i++){//printing the matrix
            for(int j=0; j<n;j++){
                System.out.print("\t"+M[i][j]);
            }System.out.println();}
        nrt.close();}}
```

Variable Description Table

Type	Name	Description
char	M	Two dimensional array to store the characters
int	i ,j	Loop variables
Int	n	Stores the order of the matrix
char	ch1 ,ch2,ch3	Stores the first, second and third characters given as input by the user respectively

Output

```
BlueJ: Terminal Window - ISC projects
Options
Enter three characters
A
C
X
Enter order of Matrix
5
    A      C      C      C      A
    C      X      X      X      C
    C      X      X      X      C
    C      X      X      X      C
    A      C      C      C      A
Enter three characters
@
?
#
Enter order of Matrix
4
    @      ?      ?      @
    ?      #      #      ?
    ?      #      #      ?
    @      ?      ?      @
```

Q4) Write a program to declare a square matrix `a[][]` of order $M \times M$, where M is a positive integer and represents rows and columns for the matrix. M should be greater than 2 and less than 10. Accept the value of M from the user. Display an appropriate message for an invalid input.

Perform the following tasks:

- a) Display the original matrix.
- b) Find the sum of the elements in each row of the matrix and display them.
- c) Find the sum of the elements in each column of the matrix and display them.
- d) Find the sum of the elements of left and right diagonals of the matrix and display them.

Example 1:

INPUT:

$M = 3$

```
1  2  3
2  4  5
3  5  6
```

OUTPUT:

```
1  2  3
2  4  5
3  5  6
```

Sum of row1 = 6

Sum of row2 = 11

Sum of row3 = 14

Sum of column1 = 6

Sum of column2 = 11

Sum of column3 = 14

Sum of the left diagonal = 11

Sum of the right diagonal = 10

Example 2:

INPUT:

$M = 4$

```
7  8  9  2
4  5  6  3
8  5  3  1
7  6  4  2
```

OUTPUT:

```
7  8  9  2
```

```
4  5  6  3
8  5  3  1
7  6  4  2
```

Sum of row1 = 26

Sum of row2 = 18

Sum of row3 = 17

Sum of row4 = 19

Sum of column1 = 26

Sum of column2 = 24

Sum of column3 = 22

Sum of the left diagonal = 17

Sum of the right diagonal = 20

Algorithm

1 START

2 Take input for the order of matrix as 'n'

3 Declare and initialise a two dimensional array as `int M[][] = new int[n][n]`. This will set order of 'M' as n X n

4 Initialise variables with names `r_sum`, `c_sum`, `l_d_sum`, `r_d_sum` (standing for row sum, column sum, left diagonal sum, right diagonal sum) as type `int` and declare their values to be zero

5 Run a for loop as `for(int i=0;i<n;i++)` and inside another nested loop as `for(int j=0;j<n;j++)`

6 Inside nested loop, take input for two dimensional array M

7 After taking input, run another nested loop as Line 5 and print the array

8 Run another nested loop as in Line 5

9 Inside the inner loop, store value of `r_sum` and `c_sum` as `r_sum+= M[i][j]` and `c_sum+= M[j][i]` respectively

10 Check if `(i==j)`, if true then store value of `l_d_sum` as `l_d_sum+= M[i][j]`

11 Check if `(i+j)==(n-1)`, if true then store value of `r_d_sum` as `r_d_sum+= M[i][j]`

12 Outside the inner loop, print values of `r_sum` and `c_sum` after every iteration and re-initialise their values with zero

13 Outside the nested loop. print the values of left diagonal sum and right diagonal sum as `l_d_sum` and `r_d_sum` respectively

14 END

Source Code

```
import java.util.Scanner;

public class Sum{
    public static void main(String args[]){
        Scanner nrt=new Scanner(System.in);
        int n,r_sum=0,c_sum=0,l_d_sum=0,r_d_sum=0;
```

```

System.out.println("Enter the order of Matrix");
n=nrt.nextInt();
int M[][]=new int[n][n];
for(int i=0;i<n;i++){
    for(int j=0;j<n;j++){
        System.out.println("Enter value at row "+i+" and column "+j);
        M[i][j]=nrt.nextInt();}
// printing the array
for(int i=0;i<n;i++){
    for(int j=0;j<n;j++){ System.out.print("\t"+M[i][j]);}
    System.out.println();}
//condition checking and finding out the sums
int i,j=0;
for(i=0;i<n;i++){
    for(j=0;j<n;j++){
        r_sum+= M[i][j];
        c_sum+= M[j][i];
        if(i==j)
            l_d_sum+= M[i][j];
        if((i+j)==(n-1))
            r_d_sum+= M[i][j];}
    System.out.println("Sum of row "+i+" is "+r_sum+"\nSum of column "+j+" is "+c_sum);
    r_sum=0;
    c_sum=0;}
System.out.println("Sum of left diagonal elements of the matrix = "+l_d_sum+"\nSum of right
diagonal elements of the matrix = "+r_d_sum);
nrt.close();}

```

Variable Description Table

Variable Name	Type	Description
n	int	Order of two dimensional array
r_sum	int	stores sum of a particular row
c_sum	int	stores sum of a particular column
l_d_sum	int	store sum of left diagonal elements
r_d_sum	int	store sum of right diagonal elements
i,j	int	loop variable
M	int	Two dimensional array

Output

```
BlueJ: Terminal Window - ISC projects
Options
Enter the order of Matrix
4
Enter value at row 0 and column 0
7
Enter value at row 0 and column 1
8
Enter value at row 0 and column 2
9
Enter value at row 0 and column 3
2
Enter value at row 1 and column 0
4
Enter value at row 1 and column 1
5
Enter value at row 1 and column 2
6
Enter value at row 1 and column 3
3
Enter value at row 2 and column 0
8
Enter value at row 2 and column 1
5
Enter value at row 2 and column 2
3
Enter value at row 2 and column 3
1
```

```
BlueJ: Terminal Window - ISC projects
Options
Enter value at row 3 and column 0
7
Enter value at row 3 and column 1
6
Enter value at row 3 and column 2
4
Enter value at row 3 and column 3
2

    7      8      9      2
    4      5      6      3
    8      5      3      1
    7      6      4      2

Sum of row 1 is =26
Sum of column 5 is 26
Sum of row 2 is =18
Sum of column 5 is 24
Sum of row 3 is =17
Sum of column 5 is 22
Sum of row 4 is =19
Sum of column 5 is 8
Sum of left diagonal elements of the matrix = 17
Sum of right diagonal elements of the matrix = 20
```

Q5)

Class name	TheString
Data members/Instance variables	
Str	to store a string
Len	integer to store the length of the string
wordcount	integer to store the number of words
Cons	integer to store the number of consonants
Member functions/Methods	
TheString	default constructor to initialize the data members
TheString(String ds)	parameterized constructor to assign str=ds
void countFreq()	To count the number of words and the number of consonants and store them in wordcount and cons respectively
void Display()	to display the original string, alongside the number of words and the number of consonants

Specify the class TheString giving the details of the constructors, void countFreq() and void Display(). Define the main() method to create an object and call the functions accordingly to enable the task.

Algorithm

1. START
2. Declare class with name TheString
3. Declare variables str(String type) and wordcount, len, cons(all three of int type)
4. Declare a default constructor and inside set str="", wordcount=0, len=0, cons=0
5. Declare a parameterised constructor and set str=(ds.trim()).toUpperCase(), where ds is the parameter passed while calling the object of TheString class
6. Set len=str.length(), wordcount=0, cons=0
7. Declare a countFreq method and declare an String array d[]=str.split(" "). This stores the words in that sentence. Set wordcount=d.length()
8. Run a for each loop to iterate through array d
9. Run a inner loop i=0; i<tmp.length() and declare a char tmp_c=tmp.charAt(i)
10. Check if tmp_c is a special character or not. If it is then continue
11. Check if tmp_c is not a vowel. If not a vowel, increase value of cons by 1
12. Declare display method and print the original string, wordcount, cons
13. Declare a main method and take a string input.
14. Create an object of class TheString and pass the user input as a parameter
15. Call TheString class's countFreq and display method

16. END

Source Code

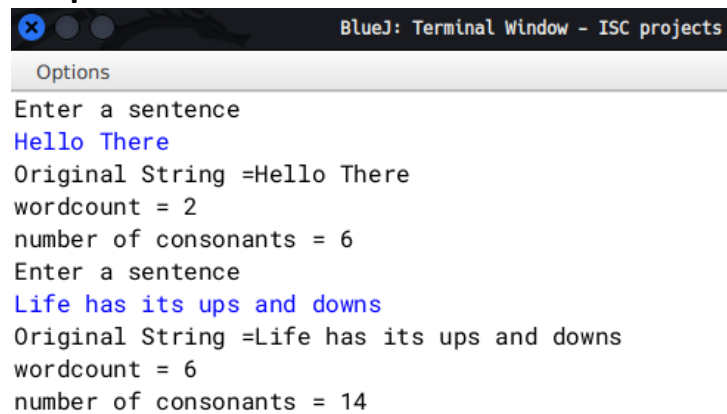
```
import java.util.Scanner;
public class TheString{
String str,copy;
int len,wordcount,cons;
TheString(){
    str="";
    wordcount=0;len=0;cons=0;}
TheString(String ds){
    copy=ds;
    str=(ds.trim()).toUpperCase();
    len=str.length();
    cons=0;wordcount=0;}
void countFreq(){
    String d[]=str.split(" ");
    wordcount=d.length;
    for(String tmp: d){
        for(int i=0;i<tmp.length();i++){
            char tmp_c=tmp.charAt(i);
            if(tmp_c=='.' || tmp_c==',' || tmp_c==' ' || tmp_c=='*' || tmp_c=='\')
                continue;
            if(tmp_c!='A' && tmp_c!='E' && tmp_c!='I' && tmp_c!='O' && tmp_c!='U')
                cons++;}}}
void display(){
    System.out.println("Original String = "+copy+"\nwordcount = "+wordcount+"\nnumber of
consonants = "+(cons));}
public static void main(){
    Scanner nrt=new Scanner(System.in);
    System.out.println("Enter a sentence");
    String user=nrt.nextLine();
    TheString obj1= new TheString(user);
    obj1.countFreq();
    obj1.display();
    nrt.close();}}
```

Variable Description Table

Variable Name	Type	Description
str	String	stores the user input
len	int	stores length of str
wordcount	int	stores the number of words in str
cons	int	stores the number of consonants in the sentence
tmp	String	loop variable

tmp_c	char	temporary character for condition checking
d	String	an array to store the words in str
user	String	stores user input in main method
copy	String	stores a copy of the original user input
obj1	class object	Object of TheString

Output



```

BlueJ: Terminal Window - ISC projects
Options
Enter a sentence
Hello There
Original String =Hello There
wordcount = 2
number of consonants = 6
Enter a sentence
Life has its ups and downs
Original String =Life has its ups and downs
wordcount = 6
number of consonants = 14

```

Q6) A Happy number is a number in which the eventual sum of the square of the digits of the number is equal to 1.

e.g. $28 = (2)^2 + (8)^2 = 4 + 64 = 68$

$68 = (6)^2 + (8)^2 = 36 + 64 = 100$

$100 = (1)^2 + (0)^2 + (0)^2 = 1 + 0 + 0 = 1$ notional

Hence, 28 is a happy number.

e.g. $12 = (1)^2 + (2)^2 = 1 + 4 = 5$

Hence, 12 is not a happy number.

Design a class Happy to check if a given number is a happy number. Some of the members of the class are given below:

Class name:	Happy
Data members/instance variables	
n	store the numbers
Member functions	
Happy ()	constructor to assign 0 to n
void getnum (int nn)	to assign the parameter value to the number n = nn
int sum_sq_digits (int x)	returns the sum of the square of the digits of the number x
void ishappy()	checks if the given number is a happy number by calling the function sum_sq_digits (int) and displays an appropriate message.

Specify the class Happy giving details of the constructor(), void getnum(int). int sum_sq_digits (int) and void ishappy(). Also define a main() function to create an object and call the methods to check for happy number.

Algorithm

1. START
2. Declare class with name Happy
3. Declare a default constructor setting value of n=0
4. Declare a getnum method to set value of n to the value the user enters
5. Declare a sum_sq_digits method to take x as an argument and find out the sum of square of x
6. Inside sum_sq_digits declare int copy=x and int sum=0
7. Run a While loop as long as copy is greater than 0. assign sum as $sum += \text{Math.pow}((\text{copy}\%10), 2)$ and $\text{copy}/=10$
8. Return sum
9. Declare a isHappy method to check if n is a Happy number or not
10. Inside declare int result=n and run a while loop as long as $\text{result}\neq 1$ and $\text{result}\neq 4$
11. Inside assign $\text{result}=\text{sum_sq_digits}(\text{result})$

12. Outside the while loop, check if result is 1 or not. If result is 1 , print n is a happy number else print n is not a happy number
13. Declare a main method to take input from user.
14. Create a object of Happy class and call getnum method and pass user input as argument and call isHappy method
15. END

Source code

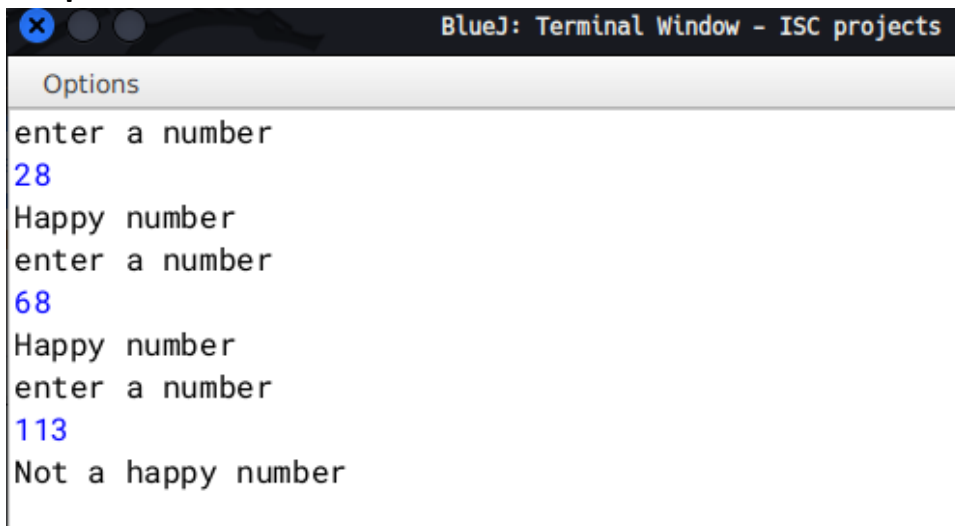
```
import java.util.Scanner;
public class Happy{
int n;
Happy(){
    n=0;}
void getnum(int nn){
    n=nn;}
int sum_sq_digits(int x){
    int copy=x,sum=0;
    while(copy>0){
        sum +=Math.pow((copy%10),2);
        copy/=10;}
    return sum;}
void isHappy(){
    int result=n;
    while(result != 1 && result != 4){
        result = sum_sq_digits(result);}
    //happy numbers end with 1
    if(result==1)
        System.out.println("Happy number");
    else if(result==4)
        System.out.println("Not a happy number");}
public static void main(){
    Scanner nrt=new Scanner(System.in);
    System.out.println("enter a number");
    int user=nrt.nextInt();
    Happy H=new Happy();
    H.getnum(user);
    H.isHappy();
    nrt.close();}}
```

Variable Description Table

Variable Name	Type	Description
n	int	to store user input
nn	int	argument for getnum method
x	int	argument for sum_sq_digits method

copy	int	stores a copy of x
sum	int	stores the sum of squares of x
result	int	stores value of n
user	int	stores user input and is required for passing as argument to getnum method

Output



```

BlueJ: Terminal Window - ISC projects
Options
enter a number
28
Happy number
enter a number
68
Happy number
enter a number
113
Not a happy number

```

Q7) Input a sentence from the user and count the number of times, the words “an” and “and” are present in the sentence. Design a class Frequency using the description given below:

Class name : Frequency

Data Members/ variables :

Text : stores the sentence

Countand : to store the frequency of the word “and”

Countan : to store the frequency of the word “an”

Len : stores the length of the string

Member functions / methods:

Frequency() :constructor to initialize the instance variables

void accept(String n):to assign n to text,where the value of the parameter n should be in lower case.

void checkandfreq() :to count the frequency of “and”

void checkanfreq() :to count the frequency of “an”

void display() :to display the number of “and” and “an” with appropriate messages.

Specify the class Frequency giving details of the constructor(), void accept(String),void checkandfreq(),void checkanfreq() and void display().Also define the main() function to create an object and call methods accordingly to enable the task.

Algorithm

1. START
2. Declare class with name Frequency
3. Declare variables text(String type) and countand,countan, len(all of int type)
4. Declare a default constructor and inside set values of text="" and 0 for len,countand,countan
5. Declare a accept method to accept the string form user and set text=n
6. Declare a checkandfreq method to count the frequency of "and"
7. Inside declare and initialise a String array d[]=text.split(" ")
8. Run a for each loop and check if tmp.equals("and")
9. Declare a checkanfreq method to count the frequency of "an"
10. Repeat step 7
11. Run a for each loop and check if tmp.equals("an")
12. Declare a display method to print the frequencies of "and" and "an"
13. Declare a main method to take input from user.
14. Create a object of Frequency class and call accept method,checkandfreq, checkanfreq,display
15. END

Source Code

```
import java.util.Scanner;
```

```

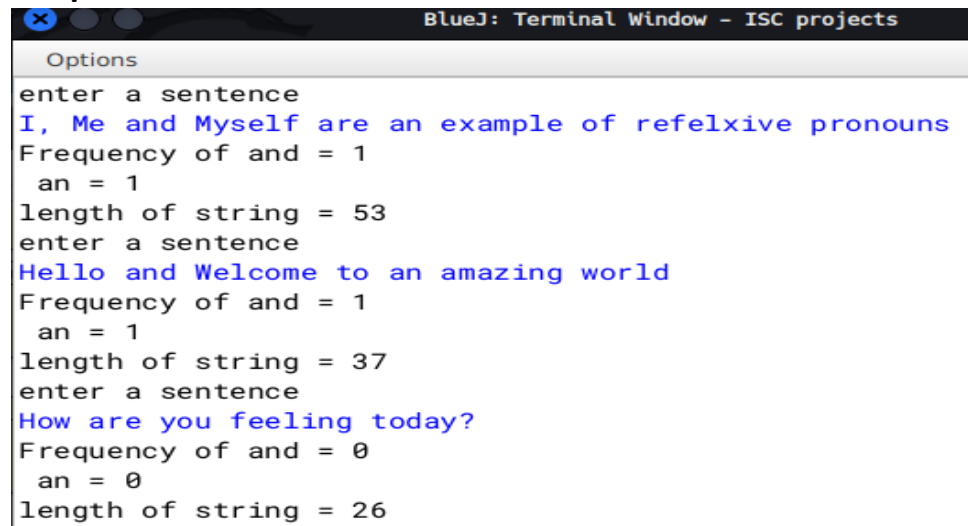
public class Frequency{
String text;
int countand,countan,len;
Frequency(){
text="";
countand=0;
countan=0;
len=0;}
void accept(String n){
text=(n.trim()).toLowerCase();
len=text.length();}
void checkandfreq(){
String d[]=text.split(" ");
for(String tmp:d){
if(tmp.equals("and"))
countand++;} }
void checkanfreq(){
String d[]=text.split(" ");
for(String tmp:d){
if(tmp.equals("an"))
countan++;} }
void display(){
System.out.println("Frequency of and = "+countand+"\n an = "+countan+"\nlength of string =
"+(text.length()));}
public void main(){
Scanner nrt=new Scanner(System.in);
System.out.println("enter a sentence");
String user=nrt.nextLine();
Frequency F=new Frequency();
F.accept(user);
F.checkandfreq();
F.checkanfreq();
F.display();
nrt.close();} }

```

Variable Description Table

Variable name	Type	Description
text	String	to store user input
len	int	to store length of text
countand	int	to store frequency of "and"
countan	int	to store frequency of "an"
tmp	String	loop variable
F	class object	
d	String array	stores the words in text

Output



A screenshot of a BlueJ Terminal Window titled "BlueJ: Terminal Window - ISC projects". The window has a standard macOS-style title bar with a red close button, a yellow maximize button, and a green full-screen button. Below the title bar is a menu bar with the word "Options" highlighted. The terminal area displays the following text:

```
enter a sentence
I, Me and Myself are an example of refelxive pronouns
Frequency of and = 1
  an = 1
length of string = 53
enter a sentence
Hello and Welcome to an amazing world
Frequency of and = 1
  an = 1
length of string = 37
enter a sentence
How are you feeling today?
Frequency of and = 0
  an = 0
length of string = 26
```


Q8) 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 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 tow matrices are equal or not. Assume that the two matrices have the same dimension.

Class name : EqMat

Data members:

a[][] : to store integer elements
m, n : to store the number of rows and columns

Member functions:

EqMat(int mm, int nn) : initialize the data members m=mm and n=nn
void readarray() : to enter the 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 and define main() to create objects and call the functions accordingly to enable the task.

Algorithm

1. START
2. Declare class with name EqMat
3. Declare two dimensional array int a[][] and int m,n;
4. Declare a parameterised constructor taking arguments mm and nn and inside set values of m=mm and n=nn;
5. Declare a readArray method to accept elements of the array by running two for loops as int i=0;i<m;i++ and inside int j=0;j<n;j++
6. Declare a check method to accept two objects of EqMat class check if two arrays are equal or not by running loops as int i=0;i<m;i++ and inside int j=0;j<n;j++ and inside checking if P.
7. Declare a print method to print the array by running two for loops as int i=0;i<m;i++ and inside int j=0;j<n;j++
8. Inside main method take user input for rows and columns and initialise two objects of EqMat class and pass rows and columns as parameters while creating them
9. class readArray and print method for both the objects and finally call check method to determine if the two matrix are equal or not
10. End

Source Code

```
import java.util.Scanner;
```

```

public class EqMat{
int a[][];
int m,n;
EqMat(int mm,int nn){
m=mm;
n=nn;
a=new int[m][n];}
void readArray(){
Scanner nrt=new Scanner(System.in);
for(int i=0;i<m;i++){
for(int j=0;j<n;j++){
System.out.println("enter element at position: "+i+" "+j);
a[i][j]=nrt.nextInt();}}}
int check(EqMat P, EqMat Q){
for(int i=0;i<m;i++){
for(int j=0;j<n;j++){
if(P.a[i][j] != Q.a[i][j])
return 0;}}
return 1;}
void print(){
for(int i=0;i<m;i++){
for(int j=0;j<n;j++){
System.out.print(a[i][j]+"\\t");}
System.out.println();}}
public static void main(){
Scanner nrt=new Scanner(System.in);
System.out.println("enter number of rows and columns");
int row=nrt.nextInt();
int col=nrt.nextInt();
EqMat A=new EqMat(row,col);
EqMat B=new EqMat(row,col);
A.readArray();
A.print();
B.readArray();
B.print();
if(A.check(A,B)==1)
System.out.println("Equal Matrix");
else
System.out.println("Not Equal Matrix");
nrt.close();}}

```

Variable Description Table

Variable name	Type	Description
i,j	int	loop variables
mm, nn	int	constructor parameter

m	int	stores number of rows
n	int	stores number of columns
P,Q	Objects of class EqMat	
a	two dimensional int array	stores the matrix

Output

```

BlueJ: Terminal Window - ISC projects
Options
enter number of rows and columns
3
enter element at position: 0 0
1
enter element at position: 0 1
2
enter element at position: 0 2
3
enter element at position: 1 0
2
enter element at position: 1 1
4
enter element at position: 1 2
5
enter element at position: 2 0
3
enter element at position: 2 1
5
enter element at position: 2 2
6
1      2      3
2      4      5
3      5      6
enter element at position: 0 0
1
enter element at position: 0 1
2
enter element at position: 0 2
3
enter element at position: 1 0
2
enter element at position: 1 1
4

```

```

enter element at position: 1 1
4
enter element at position: 1 2
5
enter element at position: 2 0
3
enter element at position: 2 1
5
enter element at position: 2 2
6
1      2      3
2      4      5
3      5      6
Equal Matrix

```

Q9) 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. 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)

Algorithm

1. START
2. Declare a class Adder
3. Declare a int array a and h_ans and m_ans both to 0
4. Declare a default constructor setting the value of array a[0]=0 and a[1]=0
5. Declare a method readtime to take input of hours and minutes
6. Declare a method addtime taking two objects(X ,Y) of class Adder as parameters and inside first check if the total sum of minutes of both the time are greater than or equal to 60 or not
7. If greater than or equal to 60 find the quotient and remainder by dividing the sum by 60 and m_sum%60 for finding the remainder respectively and setting h_ans = quotient + X.a[0] + Y.a[0] and m_ans = remainder
8. Else set h_ans = X.a[0] + Y.a[0] and m_ans= X.a[1] + Y.a[1]
9. Declare a method disptime to display the added time
10. Inside main method Declare two objects(A,B) of class Adder and call them as

```
A.readtime();
B.readtime();
A.addtime(A,B);
A.disptime();
```
11. END

Source Code

```
import java.util.Scanner;
public class Adder{
    int a[]=new int[2];
    int h_ans=0,m_ans=0;
    Adder(){
        a[0]=0;
```

```

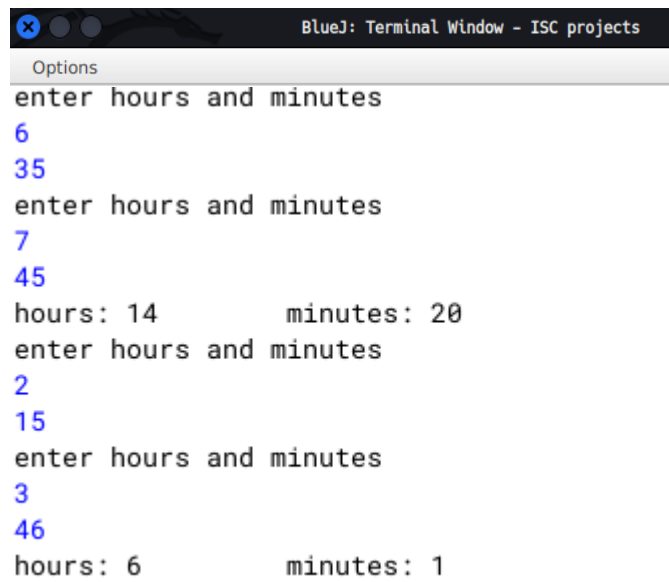
a[1]=0;}
void readtime(){
Scanner nrt=new Scanner(System.in);
System.out.println("enter hours and minutes");
a[0]=nrt.nextInt();
a[1]=nrt.nextInt();}
void addtime(Adder X, Adder Y){
int m_sum=X.a[1] + Y.a[1];
if(m_sum >= 60){
int tmp= m_sum/60;
int tmp2=m_sum%60;
m_ans=tmp2;
h_ans=(tmp+ X.a[0] + Y.a[0]);}
else{
h_ans= X.a[0] + Y.a[0];
m_ans= X.a[1] + Y.a[1];
}}
void disptime(){
System.out.println("hours: "+h_ans+"\t minutes: "+m_ans);}
public static void main(){
Scanner nrt=new Scanner(System.in);
Adder A=new Adder();
Adder B=new Adder();
A.readtime();
B.readtime();
A.addtime(A,B);
A.disptime();
nrt.close();} }

```

Variable Description Table

Variable name	Type	Description
a	one dimensional int array	stores the hours and minutes
h_ans	int	store the final hours
m_ans	int	store the final minutes
m_sum	int	stores the total minutes of both the time
tmp	int	quotient (stores the number of hours the total minutes would amount to)
tmp2	int	remainder (stores the number of minutes remaining)
A,B	objects of class Adder	

Output



```
BlueJ: Terminal Window - ISC projects
Options
enter hours and minutes
6
35
enter hours and minutes
7
45
hours: 14      minutes: 20
enter hours and minutes
2
15
enter hours and minutes
3
46
hours: 6      minutes: 1
```

Q10) Write a program to declare a matrix a[][] of order (M × N) where 'M' is the number of rows and 'N' is the number of columns such that both M and N must be greater than 2 and less than 20. Allow the user to input integers into this matrix. Perform the following tasks on the matrix:

Display the input matrix. Find the maximum and minimum value in each row and each column of the matrix and display them.

Algorithm

1. START
2. Declare a class named MaxMin
3. Declare a two dimension array as int arr[][] and initialise int M,N
4. Declare a parameterised constructor to take the rows and columns and inside set M=m; N=n and set arr=new int[m][n]
5. Declare a method readArray to take input for the array.
6. Run a loop as for(int i=0;i<M;i++) and inside for(int j=0;j<N;j++)
7. Inside take input as arr[i][j]=nrt.nextInt()
8. Declare a method display to print the array by running a loop as in LINE 6
9. Declare a method findMaxMin. Inside run a loop as for (int i=0;i<M;i++) and inside initialise int maxR=0,minR=arr[i][0]. Run another loop as for(int j=0;j<N;j++). Inside check if(maxR<arr[i][j]) then set maxR=arr[i][j]. Also check if(minR>arr[i][j]) then set minR=arr[i][j]. Outside the inner loop print the max and min value ans re-set maxR=0;minR=0.
10. Outside the loops, run another loop for columns as for(int j=0;j<N;j++) and initialize int maxC=0,minC=arr[0][j]. Run another loop as for(int i=0;i<M;i++). Inside check if(maxC<arr[i][j]) then set maxC=arr[i][j]. Also check if if(minC>arr[i][j]) then set minC=arr[i][j]. Outside the inner loop print the max and min values and re-set minC=0;maxC=0.
11. Declare a main method to take user input for number of rows and columns. Check if both rows and columns are greater than 2 and lesser than 20 or not. If not then exit showing appropriate message. Else, call the readArray , display and findMaxMin methods.
12. END

Source Code

```
import java.util.Scanner;
public class MaxMin{
    int arr[][];
    int M,N;
    MaxMin(int m,int n){
        M=m;N=n;
        arr=new int[m][n];}
    void readArray(){
        Scanner nrt=new Scanner(System.in);
        System.out.println("enter elements:");
        for(int i=0;i<M;i++){
            for(int j=0;j<N;j++){
                arr[i][j]=nrt.nextInt();}}
    void display(){
```

```

for(int i=0;i<M;i++){
for(int j=0;j<N;j++){
System.out.print("\t"+arr[i][j]);
System.out.println();}}
void findMaxMin(){
for(int i=0;i<M;i++){// for rows
int maxR=0,minR=arr[i][0];
for(int j=0;j<N;j++){
if(maxR<arr[i][j])
maxR=arr[i][j];
if(minR>arr[i][j])
minR=arr[i][j];}
System.out.println("Max of row: "+(i+1)+" is "+maxR+"\nMin is: "+minR);
maxR=0;minR=0;}
for(int j=0;j<N;j++){// for columns
int maxC=0,minC=arr[0][j];
for(int i=0;i<M;i++){
if(maxC<arr[i][j])
maxC=arr[i][j];
if(minC>arr[i][j])
minC=arr[i][j];}
System.out.println("Max of column: "+(j+1)+" is "+maxC+"\nMin is: "+minC);
minC=0;maxC=0;}}
public static void main(String h[]){
Scanner nrt=new Scanner(System.in);
System.out.println("enter dimensions of the matrix: ");
int row=nrt.nextInt();
int col=nrt.nextInt();
if(((row<2) || (row>20)) || ((col<2) || (col>20))){
System.out.println("Number of rows and columns must be greater than 2 and less than 20");
System.exit(0);}
MaxMin M=new MaxMin(row,col);
M.readArray();
M.display();
M.findMaxMin();}}

```

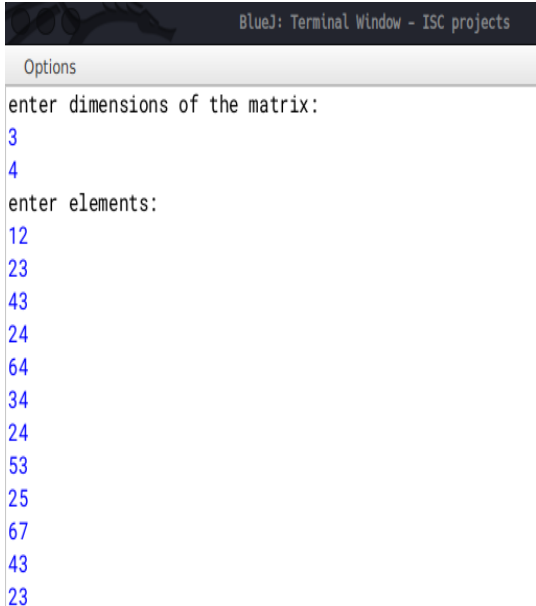
Variable Description Table

Variable	Type	Description
arr	2 dimensional integer array	stores the elements
M	int	stores the number of rows
N	int	stores the number of columns
i,j	int	loop variables
maxR	int	temporarily stores the max value for the current row

minR	int	temporarily stores the min value for the current row
maxC	int	temporarily stores the max value for the current column
minC	int	temporarily stores the min value for the current column
row	int	stores the no. of rows in main method
col	int	stores the no. of columns in main method

Output

	12	23	43	24
	64	34	24	53
	25	67	43	23



```

BlueJ: Terminal Window - ISC projects
Options
enter dimensions of the matrix:
3
4
enter elements:
12
23
43
24
64
34
24
53
25
67
43
23

```

```

Max of row: 1 is 43
Min is: 12
Max of row: 2 is 64
Min is: 24
Max of row: 3 is 67
Min is: 23
Max of column: 1 is 64
Min is: 12
Max of column: 2 is 67
Min is: 23
Max of column: 3 is 43
Min is: 24
Max of column: 4 is 53
Min is: 23

```

Q11) Design a class to overload a function stringload() as follows:

(i) void stringload (String s, char ch1, char ch2) with one string argument and two character arguments that replaces the character argument ch1 with the character argument ch2 in the given string s and prints the new string.

Example:

Input value of s ="TECHNALAGY"

ch1='A',

ch2='O'

Output : "TECHNOLOGY"

(ii) void stringload (String s) with one string argument that prints the position of the first space and the last space of the given string s.

Example:

Input value of s ="Cloud computing means Internet based computing"

Output : First index : 5

Last index : 36

(iii) void stringload (String s1, String s2) with two string arguments that combines the two strings with a space between them and prints the resultant string.

Example:

Input value of s1 ="COMMON WEALTH "

Input value of s2="GAMES "

Output : COMMON WEALTH GAMES

(use library functions)

Algorithm

1. START
2. Declare a class Overload
3. Declare a method stringload with three arguments, String s, char ch1, char ch2.
4. Declare a String ans="".
5. Run a loop as for(int i=0;i<s.length();i++) and inside initialise char tmp=s.charAt(i). Check if tmp==ch1.
6. If yes then add ch2 to ans or add tmp to ans.
7. Declare a method stringload(String s) and inside using library class print the first and last positions of space.
8. Declare a method stringload(String s1, String s2). Inside concatenate s1 and s2 and store it inside ans and print ans.
9. Declare a main method and create a object of Overload class and call stringload with three different types of parameters passed.
10. End

Source Code

```
public class Overload {  
    void stringload(String s, char ch1, char ch2){
```

```

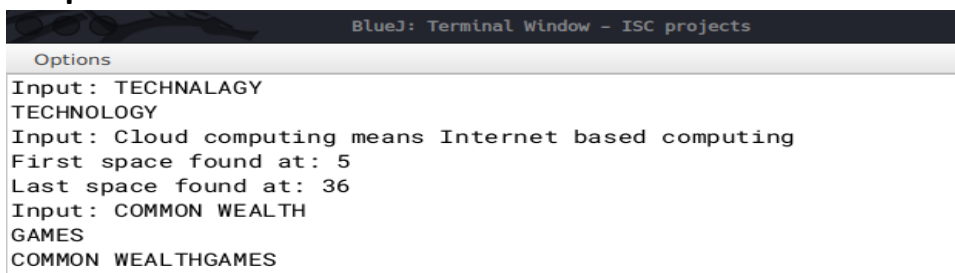
String ans="";
for(int i=0;i<s.length();i++){
    char tmp=s.charAt(i);
    if(tmp==ch1)
        ans+=ch2;
    else
        ans+=tmp;}
System.out.println(ans);}
void stringload(String s){
    System.out.println("First space found at: "+(s.indexOf(' '))+ "\nLast space found at:
"+(s.lastIndexOf(' ')));}
void stringload(String s1, String s2){
    String ans=s1+s2;
    System.out.println(ans);}
public static void main(){
    Overload O=new Overload();
    System.out.println("Input: TECHNALAGY");
    O.stringload("TECHNALAGY",'A','O');
    System.out.println("Input: Cloud computing means Internet based computing");
    O.stringload("Cloud computing means Internet based computing");
    System.out.println("Input: COMMON WEALTH \nGAMES");
    O.stringload("COMMON WEALTH","GAMES");}}

```

Variable Description Table

Variable	Type	Description
ans	String	stores the string to be showed to the user after operation
i,j	int	loop variable
tmp	char	temporarily stores the character
s, s1, s2	String	method parameters
ch1, ch2	char	method parameters

Output



```

BlueJ: Terminal Window - ISC projects
Options
Input: TECHNALAGY
TECHNOLOGY
Input: Cloud computing means Internet based computing
First space found at: 5
Last space found at: 36
Input: COMMON WEALTH
GAMES
COMMON WEALTHGAMES

```

Q12) Write a menu driven program to sort the elements of an integer array using bubble sort, selection sort, insertion sort

Algorithm

1. Start
2. Declare a class named Sorting
3. Declare a one dimensional array as a, declare int n
4. Declare a parameterised constructor and take argument as int nn
5. Declare a readArray method
6. Run a loop as for(int i=0;i<n;i++)
7. Inside take input for the array
8. Declare a print method to print the array by running a loop as in LINE 6
9. Declare a method named bubblesort. Inside initialise int temp=0. Run a loop as in LINE 6. Inside the loop run another loop as for(int j=1; j < (n-i); j++).
10. Inside the inner loop check if a[j-1] > a[j]. If yes the set temp= a[j-1] and a[j-1] = a[j] and a[j] = temp
11. Declare a method named selectionsort and inside run a loop as in LINE 6 and inside initialise int index =i.
12. Inside run another loop as for (int j = i + 1; j < a.length; j++). Inside check if a[j] < a[index]. If yes, set index=j.
13. Outer the inner loop, initialise temp= a[index] and a[index]=a[i] and a[i]=temp
14. Declare a method named insertionsort. Inside initialise int sortvalue=0. Inside set sortvalue=a[i] and initialise int j. Inside the inner loop check if a[j]>sortvalue. If yes the set a[j+1]=a[j] else break the inner loop. Outside the inner loop set a[j+1]=sortvalue.
15. Declare a main method and inside take user input in int n for length of the array. Initialise an object of class Sorting. Print the options to user. Inside a switch case call the appropriate function depending on the user's choice.
16. End

Source Code

```
import java.util.Scanner;
public class Sorting{
    int a[];
    int n;
    Sorting(int nn){
        n=nn;
        a=new int[n];}
    void readarray(){
        Scanner sc = new Scanner(System.in);
        for(int i=0;i<n;i++){
            a[i]=sc.nextInt();}}
    void print(){
        System.out.println("Array elements:");
        for(int i=0;i<n;i++){
            System.out.println(a[i]);}}
```

```

void bubblesort(){
    int temp = 0;
    for(int i=0; i < n; i++){
        for(int j=1; j < (n-i); j++){
            if(a[j-1] > a[j]){ //swap elements
                temp = a[j-1];
                a[j-1] = a[j];
                a[j] = temp; }}}
void selectionsort(){
    for (int i = 0; i < a.length - 1; i++) {
        int index = i;
        for (int j = i + 1; j < a.length; j++){
            if (a[j] < a[index]){
                index = j;//searching for lowest index  }}
        int temp= a[index];
        a[index] = a[i];
        a[i] = temp;  }}
void insertionsort(){
    int sortvalue=0;
    for(int i=1; i<a.length; i++){
        sortvalue=a[i];
        int j;
        for(j=i-1; j>=0; j--)
        {
            if(a[j]>sortvalue){a[j+1]=a[j];}
            else
                {break;}}
        a[j+1]=sortvalue;}}
public static void main(String args[]){
    Scanner sc = new Scanner(System.in);
    System.out.println("Enter the capacity of array");
    int n=sc.nextInt();
    Sorting s=new Sorting(n);
    System.out.println("Enter the elements of array");
    s.readarray();
    s.print();
    System.out.println("Select sorting technique:");
    System.out.println("1. for bubble sort");
    System.out.println("2. for selection sort");
    System.out.println("3. for insertion sort");
    int c=sc.nextInt();
    switch(c){
        case 1:
            s.bubblesort();

```

```

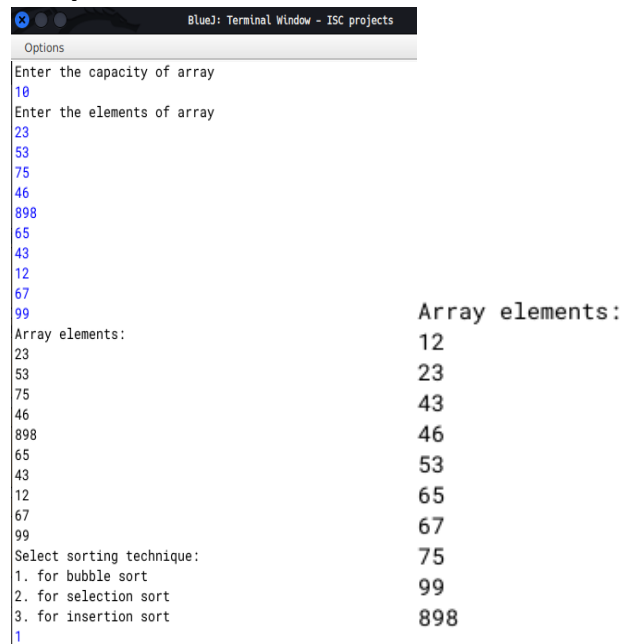
        s.print();
        break;
    case 2:
        s.selectionsort();
        s.print();
        break;
    case 3:
        s.insertionsort();
        s.print();
        break;}}}

```

Variable Description Table

Variable	Type	Description
i,j	int	loop variables
a	Integer array	stores the unsorted array
n	int	stores the length of array
temp	int	stores temporary numbers for checking
index	int	stores the index of array
sortvalue	int	stores the element to be swapped

Output



```

BlueJ: Terminal Window - ISC projects
Options
Enter the capacity of array
10
Enter the elements of array
23
53
75
46
898
65
43
12
67
99
Array elements:
23
53
75
46
898
65
43
12
67
99
Select sorting technique:
1. for bubble sort
2. for selection sort
3. for insertion sort
1

```

Enter the capacity of array	Enter the capacity of array
5	5
Enter the elements of array	Enter the elements of array
23	34
46	56
74	82
24	14
890	378
Array elements:	Array elements:
23	34
46	56
74	82
24	14
890	378
Select sorting technique:	Select sorting technique:
1. for bubble sort	1. for bubble sort
2. for selection sort	2. for selection sort
3. for insertion sort	3. for insertion sort
2	3
Array elements:	Array elements:
23	14
24	34
46	56
74	82
890	378

Q13) Write a menu driven program in java to convert a binary to decimal and vice versa

Algorithm

1. START
2. Declare a class DecToBin
3. Declare a method "binary_to_decimal" and using recursion inside check if the argument passed is 0.
4. If not zero return $n\%10 + 2 * \text{binary_to_decimal}(n/10)$.
5. If argument is zero then return 0.
6. Declare a method "decimal_to_binary" and using recursion inside check if argument passed is zero.
7. If argument is not zero return $(n\%2 + 10 * \text{decimal_to_binary}(n/2))$. If argument is zero, return 0.
8. Declare a main method print out the choice to user and input the number.
9. Check the users choice and accordingly call the methods and display the result.
10. END

Source Code

```
import java.util.Scanner;
public class DecToBin{
int binary_to_decimal(int n){
if(n==0)
return 0;
return n%10+2*binary_to_decimal(n/10);}
int decimal_to_binary(int n){
if(n==0)
return 0;
else
return (n%2 +10 * decimal_to_binary(n/2));}
public static void main(String arg[]){
Scanner nrt=new Scanner(System.in);
System.out.println("1: Binary to Decimal\n2:Decimal to Binary");
int choice=nrt.nextInt();
System.out.println("Enter the number:");
int num=nrt.nextInt();
DecToBin A=new DecToBin();
if(choice==1){
System.out.println("Decimal version of "+num+" is "+(A.binary_to_decimal(num)));}
else if (choice == 2){
System.out.println("Binary version of "+num+" is "+(A.decimal_to_binary(num)));}
nrt.close();}}
```


Variable Description Table

Variable	Type	Description
n	int	method parameter
choice	int	stores user choice
num	int	stores user input

Output

```
BlueJ: Terminal Window - ISC projects
Options
1: Binary to Decimal
2:Decimal to Binary
1
Enter the number:
1010010
Decimal version of 1010010 is 82
1: Binary to Decimal
2:Decimal to Binary
2
Enter the number:
52
Binary version of 52 is 110100
```

```
BlueJ: Terminal Window - ISC projects
Options
1: Binary to Decimal
2:Decimal to Binary
1
Enter the number:
10010
Decimal version of 10010 is 18
1: Binary to Decimal
2:Decimal to Binary
2
Enter the number:
117
Binary version of 117 is 1110101
```

Q14) Write a program to check if a number is Disarium or not

Algorithm

1. START
2. Declare a class named Disarium and initialize variables int num,size.
3. Declare a parameterised constructor taking int n as argument and inside set num=n and size=0.
4. Declare a method countDigits and inside initialize variables as int copy=num,count=0.
5. Run a while loop as long as copy is greater than 0 and post increment count. Outside the loop set size = count.
6. Declare a method SumOfDigits taking arguments as int n,int p.
7. Check if n is 0 then return 0 else return (int)(Math.pow(n%10,p))+SumOfDigits(n/10,p-1).
8. Declare a main method and take user input.
9. Initialize a object of class Disarium and call countDigits and check methods.
10. END

Source Code

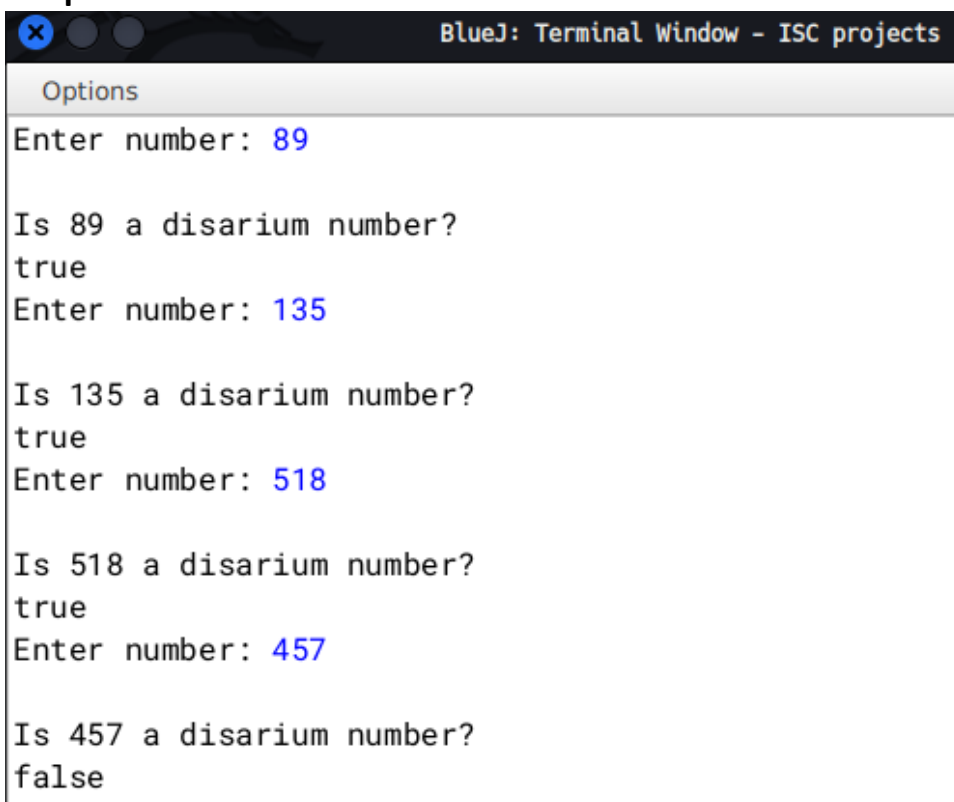
```
import java.util.Scanner;
public class Disarium{
    int num,size;
    Disarium(int n){
        num=n;
        size=0;}
    void countDigits(){
        int copy=num,count=0;
        while(copy>0){
            copy/= 10;
            count++;}
        size=count;}
    int SumOfDigits(int n, int p){
        if(n==0)
            return 0;
        else{
            return (int)(Math.pow(n%10,p))+SumOfDigits(n/10,p-1);}}
    void check(){
        int sum=SumOfDigits(num,size);
        if(sum==num)
            System.out.println(num+" is a Disarium number");
        else
            System.out.println(num+" is not a Disarium number");}
    public static void main(String h[]){
        Scanner nrt=new Scanner(System.in);
        System.out.print("Enter number: ");
        int user=nrt.nextInt();
        Disarium D=new Disarium(user);
        D.countDigits();
    }
}
```

```
D.check();  
nrt.close();}}
```

Variable Description Table

Variable name	type	Description
num	int	stores the original number
size	int	stores the length of the number
copy	int	stores the copy of num
count	int	keeps a count while counting the digits
user	int	stores user input
sum	int	stores the sum of num

Output



```
BlueJ: Terminal Window - ISC projects  
Options  
Enter number: 89  
  
Is 89 a disarium number?  
true  
Enter number: 135  
  
Is 135 a disarium number?  
true  
Enter number: 518  
  
Is 518 a disarium number?  
true  
Enter number: 457  
  
Is 457 a disarium number?  
false
```

Q15) Write a program in Java to perform Binary Search recursively. Take input from user and sort it first using any method.

Algorithm

1. Start
2. Declare a class named RecurBinSearch
3. Declare an int array as int a[];
4. Declare a take_input method to take user input.
5. Inside set a=new int[len], where len is the length of the array as given by the user
6. Run a loop as for(int i=0;i<len;i++), where len is the length of the array 'a'
7. Inside take individual input for the array.
8. Declare a method display to print the array by running a loop as in LINE6
9. Declare a method named insertionsort. Inside initialise int sortvalue=0. Inside set sortvalue=a[i] and initialise int j. Inside the inner loop check if a[j]>sortvalue. If yes the set a[j+1]=a[j] else break the inner loop. Outside the inner loop set a[j+1]=sortvalue.
10. Declare a method binary_search with return type of int, taking parameters as array, target element, start position, end position. Inside check start <= end. If true continue else return -1 and end the method there.
11. Inside the if block set int mid=(start+end)/2
12. Check if(target==a[mid]), if true, return (mid+1)
13. Else check if(target > a[mid]), if true then return binary_search(a,target, mid+1 ,end)
14. Else return binary_search(a,target, start, mid-1)
15. Declare a main method and inside ask the user for the length of the array and store in user_len. Call take_input method and display method. Then call insertionsort method and print array again using the display method. Inside the Print statement call as binary_search(A.a,t,0,((A.a).length))
- 16.End

Source Code

```
import java.util.Scanner;
public class RecurBinSearch{
int a[];
void take_input(int len){
Scanner nrt=new Scanner(System.in);
a=new int[len];
System.out.println("enter elements...");
for(int i=0;i<len;i++){
a[i]=nrt.nextInt();}}
void display(){
for(int i=0;i<a.length;i++){
System.out.print(a[i]+"\\t");}
System.out.println();}
void insertionsort(){
int sortvalue=0;
for(int i=1; i<a.length; i++){
```

```

        sortvalue=a[i];
        int j;
        for(j=i-1; j>=0; j--)
        { if(a[j]>sortvalue){a[j+1]=a[j];}
          else
            {break;}}
        a[j+1]=sortvalue;}}
    int binary_search(int a[],int target,int start,int end){
        if(start <= end){
int mid=(start+end)/2;
if(target==a[mid])
return mid+1;
else if(target > a[mid])
return binary_search(a,target, mid+1 ,end);
else
return binary_search(a,target, start, mid-1);}
return -1;}
public static void main(String s[]){
Scanner nrt=new Scanner(System.in);
System.out.println("enter length of array: ");
int user_len=nrt.nextInt();
RecurBinSearch A=new RecurBinSearch();
A.take_input(user_len);
System.out.println("Original array");
A.display();
System.out.println("enter target element");
int t=nrt.nextInt();
A.insertionsort();
System.out.println("after sorting");
A.display();
System.out.println("Element found at : "+(A.binary_search(A.a,t,0,((A.a).length)))));}

```

Variable Description Table

Variable	Type	Description
a	integer array	stores the elements in the array
i,j	int	loop variables
sortvalue	int	stores the value temporarily that is to be swapped
start	int	stores the start index
target	int	stores the element that is to be searched for
end	int	stores the end index
user_len	int	stores the length of the array

Output

```
BlueJ: Terminal Window - ISC projects
Options
enter length of array:
5
enter elements...
12
54
2
57
35
Original array
12    54    2    57    35
enter target element
57
after sorting
2    12    35    54    57
Element found at : 5
enter length of array:
4
enter elements...
32
13
67
43
Original array
32    13    67    43
enter target element
32
after sorting
13    32    43    67
Element found at : 2
```

Q16) Write a Java Program to perform the operation of adding, subtracting and multiplying two matrices

Algorithm

1. START
2. Declare a class named Matrix
3. Initialize values as int m1_r,m1_c,m2_r,m2_c and boolean sum_difference=false,product=false
4. Declare a method as take_input to take input of the rows and columns of both the matrices
5. Declare a method check to check if sum, difference and product Operation is possible or not. If both are impossible then exit showing appropriate message
6. Declare a method takeElementInput to take the input of elements of both the matrices
7. Declare a method display to print the matrices by running a loop as for(int i=0;i<m2_r;i++) and inside for(int j=0;j<m2_c;j++)
8. Declare a method sum and check if both their rows and columns are same or not. Then run a loop as for(int i=0;i<m2_r;i++) and put the result in the ans matrix.
9. Declare a method difference and check if both their rows and columns are same or not. Then run a loop as for(int i=0;i<m2_r;i++) and put the result in the ans matrix.
10. Declare a method find_product and run a loop as for(int i=0;i<m1_r;i++) and inside as for(int j=0;j<m2_c;j++). Inside set ans[i][j]=0. Run a third loop as for(int k=0;k<m1_c;k++) and inside set ans[i][j] += (M1[i][k]*M2[k][j]). Outside the three loops print the ans matrix.
11. Declare a main method and call the methods accordingly.
12. END

Source Code

```
import java.util.Scanner;
public class Matrix{
    int M1[][],M2[][];
    int m1_r,m1_c,m2_r,m2_c;
    boolean sum_difference=false,product=false;
    void take_input(){
        System.out.println("Enter no. of rows & columns for the first matrix");
        Scanner nrt=new Scanner(System.in);
        m1_r=nrt.nextInt();
        m1_c=nrt.nextInt();
        M1=new int[m1_r][m1_c];
        System.out.println("Enter no. of rows & columns for the second matrix");
        m2_r=nrt.nextInt();
        m2_c=nrt.nextInt();
        M2= new int[m2_r][m2_c];
    }
    void check(){
        if((m1_r==m2_r)&&(m1_c==m2_c))
            sum_difference=true;
        if(m1_c==m2_r)
            product=true;
        if(sum_difference==false && product==false){
            System.out.println("Operation of sum, difference, product are not possible.\nExiting...");
        }
    }
}
```

```

System.exit(0);}}
void takeElementInput(){
Scanner nrt=new Scanner(System.in);
System.out.println("Enter elements for the first matrix");
for(int i=0;i<m1_r;i++){
for(int j=0;j<m1_c;j++){
M1[i][j]=nrt.nextInt();}}
System.out.println("Enter elements for the second matrix");
for(int i=0;i<m2_r;i++){
for(int j=0;j<m2_c;j++){
M2[i][j]=nrt.nextInt();}}}
void display(){
System.out.println("-----");
for(int i=0;i<m1_r;i++){
for(int j=0;j<m1_c;j++){
System.out.print(M1[i][j]+"\\t");
}System.out.println();
}
System.out.println("-----");
for(int i=0;i<m2_r;i++){
for(int j=0;j<m2_c;j++){
System.out.print(M2[i][j]+"\\t");
}System.out.println();
}
System.out.println("-----");
}
void sum(){
if(sum_difference==true){int ans[][]=new int[m1_r][m1_c];
for(int i=0;i<m1_r;i++){
for(int j=0;j<m1_c;j++){
ans[i][j]=(M1[i][j]+M2[i][j]);
System.out.print(ans[i][j]+"\\t");
}System.out.println();
}}
else
System.out.println("Not Possible");
}
void difference(){
if(sum_difference==true){
int ans[][]=new int[m1_r][m1_c];
for(int i=0;i<m1_r;i++){
for(int j=0;j<m1_c;j++){
ans[i][j]=(M1[i][j]-M2[i][j]);
System.out.print(ans[i][j]+"\\t");
}System.out.println();
}}
else
System.out.println("Not Possible");
}

```



```

void find_product(){
int ans[][]=new int[m1_r][m2_c];
for(int i=0;i<m1_r;i++){
for(int j=0;j<m2_c;j++){
ans[i][j]=0;
for(int k=0;k<m1_c;k++){
ans[i][j] += (M1[i][k]*M2[k][j]);}}}
//displaying...
for(int i=0;i<m1_r;i++){
for(int j=0;j<m2_c;j++){
System.out.print(ans[i][j]+"\\t");
}System.out.println();}}
public static void main(String args[]){
Matrix O=new Matrix();
O.take_input();
O.check();
O.takeElementInput();
O.display();
Scanner nrt=new Scanner(System.in);
System.out.println("1:Find Sum\\n2:Find Difference\\n3:Find Product");
switch(nrt.nextInt()){
case 1:
O.sum();
break;
case 2:
O.difference();
break;
case 3:
O.find_product();
break;
default:
System.out.println("Invalid Input");}}}

```

Variable Description Table

Variable	Type	Description
M1	two-dimensional integer matrix	stores the first matrix
M2	two-dimensional integer matrix	stores the first matrix
m1_r	int	stores the no. of rows of the 1 st matrix
m1_c	int	stores the no. of columns of the 1 st matrix
m2_r	int	stores the no. of rows of the 2 nd matrix
m2_c	int	stores the no. of columns of the 2 nd matrix
i,j	int	loop variables

sum_difference	boolean	stores true if the sum and difference operation is possible else false
product	boolean	stores true if the product operation is possible else false

Output

BlueJ: Terminal Window - ISC projects

Options

Enter no. of rows & columns for the first matrix
2
2
Enter no. of rows & columns for the second matrix
2
2
Enter elements for the first matrix
1
2
3
4
Enter elements for the second matrix
5
6
7
8

1 2
3 4

5 6
7 8

BlueJ: Terminal Window - ISC projects

Options

Enter no. of rows & columns for the first matrix
3
2
Enter no. of rows & columns for the second matrix
1
2
Operation of sum, difference, product are not possible.
Exiting...

1:Find Sum
2:Find Difference
3:Find Product
1
6 8
10 12

BlueJ: Terminal Window - ISC projects

Options

Enter no. of rows & columns for the first matrix
2
2
Enter no. of rows & columns for the second matrix
2
2
Enter elements for the first matrix
8
7
6
5
Enter elements for the second matrix
4
3
2
1

8 7
6 5

4 3
2 1

1:Find Sum
2:Find Difference
3:Find Product
2
4 4
4 4

BlueJ: Terminal Window - ISC projects

Options

Enter no. of rows & columns for the first matrix
2
2
Enter no. of rows & columns for the second matrix
2
2
Enter elements for the first matrix
4
3
5
2
Enter elements for the second matrix
1
4
6
2

4 3
5 2

1 4
6 2

1:Find Sum
2:Find Difference
3:Find Product
3
22 22
17 24

Q17) Design a program that accepts your DOB in dd/mm/yyyy format. Check if the date entered is valid or not. If valid display "Valid Date" also compute the day no. of the year for the DOB, else display "Invalid Date".

Algorithm

1. START
2. Declare a class DateChecking
3. Initialize int dd,mm,yy boolean leap=false,ch=false
4. Declare a method take_input to take date,month and year as user input
5. Declare a method checkValidity and first check if dd<32 and dd>0.
6. Then check dds value according to the month. Also check if the year is a leap year or not.
7. Then finally check for February.
8. Declare a method calculateDay and initialize int count=0.
9. Run a loop as for(int i=1; i<mm;i++) and inside according to the month add 31 or 30 days to the count. Return (count-1) as 1 day is extra.
10. Declare a main method and inside take input. If the date is valid, print "Valid Date" and call calculateDay method else print "Invalid Date"
11. END

Source Code

```
import java.util.Scanner;
public class DateChecking{
    int dd,mm,yy;
    boolean leap=false,ch=false;
    void take_input(){
        System.out.println("Enter your Date of birth in dd/mm/yyyy format...");
        Scanner nrt=new Scanner(System.in);
        dd=nrt.nextInt();
        mm=nrt.nextInt();
        yy=nrt.nextInt();
    }
    boolean checkValidity(){
        if(dd>0 && dd<32){//checking date
            if(mm==1 || mm==3 || mm==5 || mm==7 || mm==8 || mm==10 ||
mm==12){//checking months with 31 days
                if(dd<32)
                    ch=true;
            }
            if(mm==4 || mm==6 || mm==9 || mm==11){//checking months with 30 days
                if (dd<31)
                    ch=true;
            }
        }
        if (((yy % 4 == 0) && (yy % 100 != 0)) || (yy%400 == 0)){//checks leap year
            leap=true;
        }
        if(mm==2){//checks february
```

```

        if(leap==true && dd<=29)
            ch=true;
        else if(dd<=28)
            ch=true;
    }
    return ch;
}
int calculateDay(){
    int count=0;
    for(int i=1; i<mm;i++){
        if(i==1 || i==3 || i==5 || i==7 || i==8 || i==10 || i==12)
            count+=31;
        if(i==4 || i==6 || i==9 || i==11)
            count +=30;
        if(i==2){
            if(leap==true)
                count+=29;
            else
                count+=28;
        }
    }
    count+=dd;
    return count-1;
}
public static void main(String args[]){
    DateChecking D=new DateChecking();
    D.take_input();
    if(D.checkValidity())
        System.out.println("Valid Date\nDay no. of the year is: "+D.calculateDay());
    else
        System.out.println("Invalid Date");
}
}

```

Variable Description Table

Variable	Type	Description
dd	int	Stores day number
mm	int	Stores month number
yy	int	Stores Year number
leap	boolean	Stores if 'yy' is a leap year or not
ch	boolean	Temporarily stores if the date is valid or not
count	int	Stores the no. of days from the start of the year

Output

```
BlueJ: Terminal Window - ISC projects
Options
Enter your Date of birth in dd/mm/yyyy format...
14
10
2005
Valid Date
Day no. of the year is: 286
Enter your Date of birth in dd/mm/yyyy format...
23
5
2018
Valid Date
Day no. of the year is: 142
Enter your Date of birth in dd/mm/yyyy format...
29
2
2021
Invalid Date
```

Q18) A line on a plane can be represented by coordinates of the two end points p1 and p2 as p1(x1, y1) and p2(x2, y2).

A super class Plane is defined to represent a line and a subclass Circle to find the length of the radius and the area of circle by using the required data members of super class.

Some of the members of both the classes are given below:

Class name: Plane

Data members/instance variables:

x1: to store the x-coordinate of the first end point. y1: to store the y-coordinate of the first end point.

Member functions/methods:

Plane(int x, int y): parameterized constructor to assign the data members x1 = x and y1 = y. void show(): to display the coordinates.

Class name: Circle

Data members/instance variables:

x2: to store the x-coordinate of the second end point. y2: to store the y-coordinate of the second end point.

radius: double variable to store the radius of the circle.

area: double variable to store the area of the circle.

Member functions/methods:

Circle(...): parameterized constructor to assign values to data members of both the classes.

void findRadius(): to calculate the length of radius using the formula:

$$\sqrt{((x2 - x1)^2 + (y2 - y1)^2)} / 2$$
 assuming that x1, x2, y1, y2 are the coordinates of the two ends of the diameter of a circle. void findArea(): to find the area of circle using formula: πr^2 . The value of pie (π) is 22 / 7 or 3.14.

void show(): to display both the coordinates along with the length of the radius and area of the circle. Specify

the class Plane giving details of the constructor and

void show(). Using the concept of inheritance, specify

the class Circle giving details of the constructor, void

findRadius(), void findArea() void show(). Also write the main() function.

Algorithm

1. START
2. Declare a class Plane
 - i) Initialize integer values as x1,y1.
 - ii) Declare a parameterised constructor taking int x, int y as parameters.
 - iii) Set x1=x and y1=y.
 - iv) Declare a method show to print the values of x1 and x2
3. Declare a class Circle which extends Plane
 - A)
 - i) Initialize integer values as x2,y2.
 - ii) Declare a parameterised constructor taking int a, int b, int c, int d as parameters.
 - iii) Call the super class's constructor as super(a,b).
 - iv) Set x2=c, y2=d.
 - B) Declare a method named findRadius.
Set radius as $(\text{Math.sqrt}(\text{Math.pow}((x2-x1),2) + \text{Math.pow}((y2-y1),2)))/2$.
 - C) Declare a method named findArea.
Set area as $((22/7)*(\text{Math.pow}(\text{radius},2)))$.
 - D) Declare a method as show.
Print the coordinates of both classes along with the radius and area.
 - E) Declare a main method.
 - i) Take input from user for the four co-ordinates.
 - ii) Initialize a object of class Circle.
 - iii) Call findRadius, findArea, show
4. END

Source Code

//super class

```
class Plane{
    int x1,y1;
    Plane(int x, int y){
        x1=x;
        y1=y;
    }
    void show(){
        System.out.println("The coordinates(x1, y1) are: "+x1+" "+y1);
    }
}
```

//subclass

```
import java.util.*;
class Circle extends Plane{
```



```

int x2,y2;
double radius, area;
Circle(int a, int b, int c, int d){
    super(a,b);
    x2=c;
    y2=d;
}
void findRadius(){
    //System.out.println("r init"+x1+" "+y1+" "+x2+" "+y2);
    radius=(Math.sqrt( ( Math.pow((x2-x1),2) + Math.pow( (y2-y1), 2) ) ) ) /2;
    //System.out.println("r"+radius);
}
void findArea(){
    System.out.println("r init"+radius);
    area=((3.14)*(radius*radius));
}
void show(){
    super.show();
    System.out.println("The coordinates(x2, y2) are: "+x2+" "+y2);
    System.out.println("Radius: "+radius+"\nArea: "+area);
}

public void main(){
    System.out.println("Enter the 4 coordinates");
    Scanner nrt=new Scanner(System.in);
    Circle O=new Circle(nrt.nextInt(), nrt.nextInt(), nrt.nextInt(), nrt.nextInt());

    O.findRadius();
    O.findArea();
    O.show();
    nrt.close();
}
}

```

Variable Description Table

Variable Name	Data type	Description
x1	int	Stores x coordinate in the super class
y1	int	Stores y coordinate in the super class
x2	int	Stores x coordinate in the subclass
y2	int	Stores y coordinate in the subclass
radius	double	Stores the radius of the circle
area	double	Stores the area of the circle

Output

```
BlueJ: Terminal Window - TMP
Options
Enter the 4 coordinates
2
4
3
5
The coordinates(x1, y1) are: 2 4
The coordinates(x2, y2) are: 3 5
Radius: 0.7071067811865476 units
Area: 1.5700000000000005 unit square
Enter the 4 coordinates
6
7
4
8
The coordinates(x1, y1) are: 6 7
The coordinates(x2, y2) are: 4 8
Radius: 1.118033988749895 units
Area: 3.9250000000000007 unit square
```

Q19) A class whrl contains a 2-D array of order MXN. Write a java program to perform a right circular shift and a left circular shift on alternate rows and columns.

Eg: INPUT

m=5

n=4

MATRIX IS:

```
20 2 24 25
48 18 13 47
43 18 30 32
22 45 30 38
48 12 18 37
```

OUTPUT:

```
25 20 2 24
18 13 47 48
32 43 18 30
45 30 38 22
37 48 12 18
```

Algorithm

1. START
2. Declare a Class named Whrl.
3. Inside initialize integer variables as m,n, arr[][].
4. Declare a method named take_input.
 - A. Initialize a Scanner object and take input for values of m and n.
 - B. Run a loop from 0 to m
 - C. Run another loop inside from 0 to n and take the elements of the array as input
5. Declare a method named Shift.
 - A. Initialize integer variables as j,i,tmp.
 - B. Run a loop from 0 to m. Check if i is divisible by 2
 - i) If divisible, set tmp as arr[i][n-1]. Run a loop from (n-1) to 1. Inside update arr[i][j]=arr[i][j-1]. Outside the loop set arr[i][0]=tmp.
 - ii) Else, set tmp as tmp=arr[i][0]. Run a loop from 1 to n. Inside update arr[i][j-1]=arr[i][j]. Outside set arr[i][j-1]=tmp.
6. Declare a method named display.
 - i) Run a loop as 4,A and 4,B
 - ii) Inside print the values of the array.

iii) Outside the innermost loop go to a new line.

7. Declare a main method.

i) Initialize an object of class Whrl.

ii) Call methods accordingly.

Source Code

```
import java.util.*;
public class Whrl{
    int m,n;
    int arr[][];
    void take_input(){
        Scanner nrt=new Scanner(System.in);
        System.out.println("Enter the number of rows and columns");
        m=nrt.nextInt();
        n=nrt.nextInt();
        arr=new int[m][n];
        System.out.println("Enter the elements");
        for(int i=0;i<m;i++){
            for(int j=0;j<n;j++){
                arr[i][j]=nrt.nextInt();
            }
        }
    }
    void Shift(){
        int j,i,tmp;
        for(i=0;i<m;i++){
            if(i%2==0){
                tmp=arr[i][n-1];
                for(j=n-1;j>=1;j--){
                    arr[i][j]=arr[i][j-1];
                }
                arr[i][0]=tmp;
            }else{
                tmp=arr[i][0];
                for(j=1;j<n;j++){
                    arr[i][j-1]=arr[i][j];
                }
                arr[i][j-1]=tmp;
            }
        }
    }
    void display(){
        System.out.println("-----");
        for(int i=0;i<m;i++){
            for(int j=0;j<n;j++){
```

```

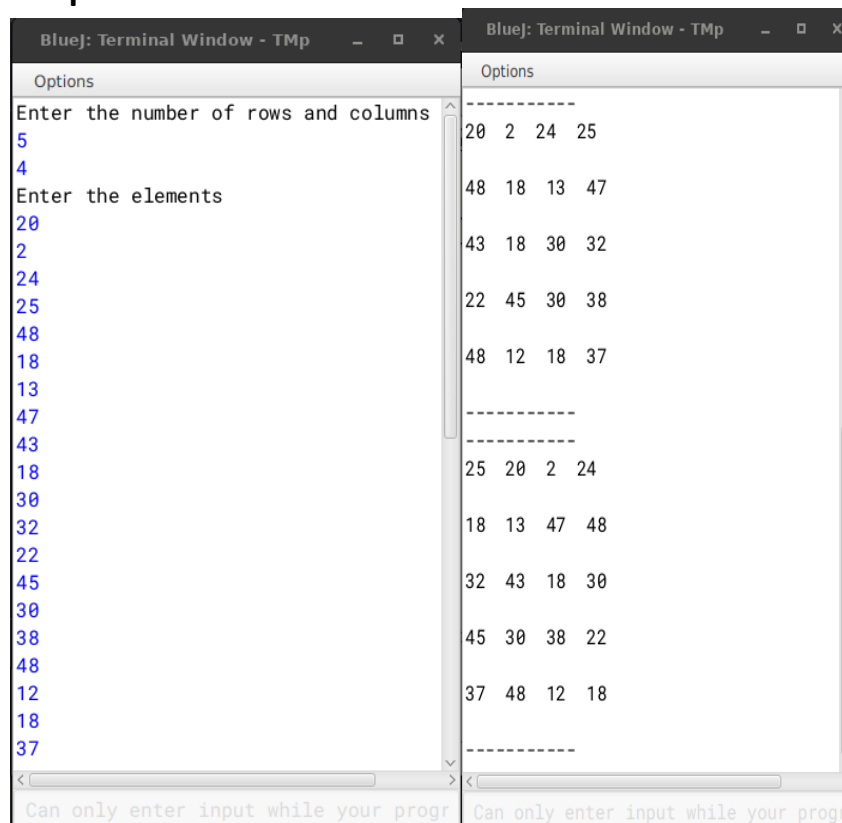
System.out.print(arr[i][j]+" ");
}
System.out.println();
System.out.println();
}
System.out.println("-----");
}
public static void main(String[] args) {
    Whrl O=new Whrl();
    O.take_input();
    O.display();
    O.Shift();
    O.display();
}
}

```

Variable Description Table

Variable Name	Data type	Description
i,j	int	loop variables
m	int	Stores number of rows
n	int	Stores number of rows
arr	int array	Stores the elements

Output



Q20) Write a menu driven java program to find the sum of the series given below.

$S=1 + (x+2)/2! + (2x+3)/3! + (3x+4)/4! + \dots n \text{ terms}$

$S=x^2/1! + x^4/3! + x^6/5! + \dots x^n/(n-1)!$

Algorithm

1. START
2. Declare a class named Series
3. Declare a method as factorial taking a int variable "a" as parameter.
4. Inside initialize int f=1. Run a for loop as for(int i=1;i<=a;i++)
5. Inside the loop update f as f*=i;
6. Declare a method as Series1 taking (int x, int n) as parameter.
7. Inside set int sum=1, and run a for loop as for(int i=1;i<=n;i++).
8. Inside the loop, update sum as sum+=((i*x)+(i+1))/(O.factorial(i+1)). Outside return sum.
9. Declare a method as Series2 taking (int x, int n) as parameter.
10. Inside set int sum=0, and run a for loop as for(int i=2;i<=n;i=i+2).
11. Inside the loop, update sum as sum+=((Math.pow(x,i))/(S.factorial(i-1))).
12. Create a main method and call the methods accordingly.
13. END

Source Code

```
import java.util.Scanner;
public class Series{
    int factorial(int a){
        int f=1;
        for(int i=1;i<=a;i++)
            f*=i;
        return f;
    }
    int Series1(int x, int n){
        Series O=new Series();
        int sum=1;
        for(int i=1; i<=n;i++){
            sum+=((i*x)+(i+1))/(O.factorial(i+1));
        }
        return sum;
    }
    int Series2(int x, int n){
        Series S=new Series();
        int sum=0;
        for(int i=2;i<=n;i=i+2){
            sum+=((Math.pow(x,i))/(S.factorial(i-1)));
        }
        return sum;
    }
    public static void main(String r[]){
        Scanner nrt=new Scanner(System.in);
```

```

        Series Obj=new Series();
        System.out.println("Enter value of x and n");
        int user_x=nrt.nextInt();
        int user_n=nrt.nextInt();
        System.out.println("Enter 1 for the first series or any other number for series 2");
        int user_choice=nrt.nextInt();
        if(user_choice==1)
            System.out.println("result = "+Obj.Series1(user_x,user_n));
        else
            System.out.println("result = "+Obj.Series2(user_x,user_n));

        nrt.close();
    }
}

```

Variable Description Table

Variable Name	Data Type	Description
f	int	Stores the factorial of a number
i	int	loop variable
sum	int	Stores the sum of the respective series
user_x	int	Stores the value of x
user_n	int	Stores the value of n
user_choice	int	Stores the user's choice

Output

```

BlueJ: Terminal Window - TMp
Options
Enter value of x and n
5
3
Enter 1 for the first series or any other number for series 2
1
result = 6
Enter value of x and n
3
3
Enter 1 for the first series or any other number for series 2
2
result = 9
Enter value of x and n
3
3
Enter 1 for the first series or any other number for series 2
1
result = 4
Enter value of x and n
6
3
Enter 1 for the first series or any other number for series 2
2
result = 36
Can only enter input while your program is running

```

Q21) An ISBN (International Standard Book Number) is a ten digit code which uniquely identifies a book. The first nine digits represent the Group, Publisher and the Title of the book and the last digit is used to check whether ISBN is correct or not. Each of the first nine digits of the code can take a value between 0 and 9. Sometimes it is necessary to make the last digit equal to ten; this is done by writing the last digit of the code as X. To verify an ISBN, calculate 10 times the first digit, plus 9 times the second digit, plus 8 times the third digit and so on until we add 1 time the last digit. If the final number leaves no remainder when divided by 11, the code is a valid ISBN.

For Example:

1. $0201103311 = 10*0 + 9*2 + 8*0 + 7*1 + 6*1 + 5*0 + 4*3 + 3*3 + 2*1 + 1*1 = 55$
Since 55 leaves no remainder when divided by 11, hence it is a valid ISBN.

2. $007462542X = 10*0 + 9*0 + 8*7 + 7*4 + 6*6 + 5*2 + 4*5 + 3*4 + 2*2 + 1*10 = 176$

Since 176 leaves no remainder when divided by 11, hence it is a valid ISBN.

3. $0112112425 = 10*0 + 9*1 + 8*1 + 7*2 + 6*1 + 5*1 + 4*1 + 3*4 + 2*2 + 1*5 = 71$
Since 71 leaves no remainder when divided by 11, hence it is not a valid ISBN.

Design a program to accept a ten digit code from the user. For an invalid input, display an appropriate message. Verify the code for its validity in the format specified below:

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

Example 1

INPUT CODE: 0201530821

OUTPUT : SUM = 99

LEAVES NO REMAINDER – VALID ISBN CODE

Example 2

INPUT CODE: 035680324

OUTPUT : INVALID INPUT

Example 3

INPUT CODE: 0231428031

OUTPUT : SUM = 122

LEAVES REMAINDER – INVALID ISBN CODE

Algorithm

1. START
2. Declare a class named ISBN
3. Declare a method named isValid returning boolean value
4. Inside check if length of the isbn provided by the user is 10 or not
if not 10, return false.
5. If length is 10, initialize int sum=0, count=10, boolean contains_x=false, int digit.
6. Check if the last character is "X" or not and accordingly update value of contains_x.
7. Run a for loop as for(int i=0; i<=9 ;i++). Check if the loop is on its last iteration. If true, then set digit as 10, else set digit as s.charAt(i)-'0'. Update sum as sum +=(count-- * digit).
8. Check if the sum is divisible by 11 or not and accordingly return a boolean value.
9. Declare a main method.
10. Take user input in the form of string
11. Initialize boolean ans=O.isValid(String.valueOf(user))
12. Print "VALID" or "INVALID" according to isValid method
13. END

Source Code

```
import java.util.*;
public class ISBN{
    boolean isValid(String s){

        if(s.length()!=10)
            return false;

        int sum=0, count=10,digit;
        boolean contains_x=false;

        if((s.charAt(s.length()-1))=='X'){
            contains_x=true;
        }

        for(int i=0; i<=9 ;i++){
            if(i==9&&contains_x==true){
                digit=10;
            }else{
                digit=s.charAt(i)-'0';
            }

            sum +=(count-- * digit) ;
        }
        System.out.println("SUM = "+sum);
        if(sum%11==0)
            return true;

        return false;
    }
}
```

```

    }
    public static void main(String f[]){
        Scanner nrt=new Scanner(System.in);
        System.out.println("Enter the isbn no.");
        String user=nrt.nextLine();
        ISBN O=new ISBN();
        boolean ans=O.isValid(String.valueOf(user));
        System.out.println(ans?("LEAVES NO REMAINDER - VALID ISBN CODE"):"LEAVES
REMAINDER - INVALID ISBN CODE");
        nrt.close();
    }
}

```

Variable Description Table

Variable Name	Data Type	Description
sum	int	Stores the sum
count	int	Keeps a count backwards and is used in getting the sum of the ISBN number
digit	int	Stores the temporary digit
i	int	loop variable
conatins_x	boolean	Stores boolean value based on if the number contains "x" at the end
user	int	Stores user input in main

Output

```
BlueJ: Terminal Window - TMP
Options
Enter the isbn no.
0201103311
SUM = 55
LEAVES NO REMAINDER - VALID ISBN CODE
Enter the isbn no.
007462542X
SUM = 176
LEAVES NO REMAINDER - VALID ISBN CODE
Enter the isbn no.
0112112425
SUM = 71
LEAVES REMAINDER - INVALID ISBN CODE
```

Q22) A superclass Number is defined to calculate the factorial of a number. Define a subclass Series to find the sum of the series $S = 1! + 2! + 3! + 4! + \dots + n!$.

The details of the members of both classes are given below:

Class name: Number

Data member/instance variable:

n: to store an integer number

Member functions/methods:

Number(): constructor to initialize the data member

int factorial(int a): returns the factorial of a number

(factorial of $n = 1 \times 2 \times 3 \times \dots \times n$)

void display() : display the member variable

Class name: Series

Data member/instance variable:

sum: to store the sum of the series

Member functions/methods:

Series(...) : parameterized constructor to initialize the data members of both the classes

void calSum(): calculates the sum of the given series

void display(): displays the data members of both the classes

Using the concept of inheritance, specify both the classes giving the details of the constructor (...), void calSum(), and void display(). Also define the main method.

Algorithm

1. START
2. Declare a class named Number
 - a) Declare a int named n.
 - b) Declare a parameterised constructor and assign $n=x$.
 - c) Declare a int method named factorial taking a int as argument.
 - d) Inside set $int\ f=1$.
 - e) Run a loop from $i=1$ to argument. Inside update f as $f*=i$.
 - f) Declare a void method named display and display the value of n.
3. Declare a class named Series which extends Number
 - a) Declare a parameterised constructor and call superclass's constructor.
 - b) Declare a void method named calSum.
 - c) Run a loop from $i=1$ to n and inside update $sum+=factorial(i)$.
 - d) Declare a void method named display and call superclass's display method and print the sum
4. Declare a main method and Initialize an object and call methods accordingly

Source Code

```

//super class
class Number{
    int n;
    Number(int x){
        n=x;
    }
    int factorial(int a){
        int f=1;
        for(int i=1;i<=a;i++)
            f*=i;
        return f;
    }
    void display(){
        System.out.println("Value of n: "+n);
    }
}

```

// subclass

```

import java.util.*;
class Series extends Number {
    int sum;
    Series(int a){
        super(a);
        sum=0;
    }
    void calSum(){
        for(int i=1;i<=n;i++){
            sum+= factorial(i);
        }
    }
    void display(){
        super.display();
        System.out.println("Sum: "+sum);
    }
    public void main(){
        System.out.println("Enter value of n");
        Scanner nrt=new Scanner(System.in);
        Series S=new Series(nrt.nextInt());
        S.calSum();
        S.display();
    }
}

```

Variable Description Table

Variable Name	Data Type	Description
n	int	Stores user input
f	int	Stores the factorial of a number
sum	int	Stores the sum of the series
i	int	loop variable

Output

```
BlueJ: Terminal Window -...
Options
Enter value of n
3
Value of n: 3
Sum: 9
Enter value of n
5
Value of n: 5
Sum: 153
Enter value of n
7
Value of n: 7
Sum: 5913
Can only enter input while your pr...
```

Q23) Wordpile is an entity which can hold maximum of 20 characters. The restriction is that a character can be added or removed from one end only.

Data members:

ch[]: character array to hold the character elements

capacity: integer variable to store the maximum capacity

top: to point to the index of the topmost element

Member functions/methods:

WordPile (int cap): constructor to initialise the data member capacity = cap,

top = -1 and create the WordPile

void pushChar(char v): adds the character to the top of WordPile if possible, otherwise output a message "WordPile is full"

char popChar(): returns the deleted character from the top of the WordPile if possible, otherwise it returns '\\'

void display(): to display the chaacters in Wordpile.

Algorithm

1. Start
2. Declare class WordPile
3. Declare a character array as char ch[]; and declare int top, capacity;
4. Declare a constructor with parameter int cap to take capacity of the stack and set top=-1, capacity=cap and ch =new char[cap];
5. Declare a pushChar method taking char v as argument and inside if stack is not already full, set ch[++top]=v else display WordPile is full.
6. Declare a popChar method to remove the topmost element of the stack. Inside if the stack is empty display else set int x=ch[top--]; and return x
7. Declare a display method to print the stack.
8. Inside run a loop as for(int i=0;i<=top;i++)
9. Inside the loop print as ch[i];
10. Declare main method and take the capacity of the stack as input. Create an object of class WordPile as W and call W.take_input(); Declare char tmp.
11. Inside a while loop take user input according to their choice to either display the stack , add character to the stack, remove the topmost character, or to exit the program using switch case and calling W.pushChar() and W.popChar() and W.display() accordingly.

12. END

Source Code

```
import java. util. Scanner;

class WordPile {

char ch[];

int capacity;

int top;

void display(){

for(int i=0;i<=top;i++){

System.out.println("\t"+ch[i]+"\\t");

}

}

public WordPile(int cap) {

capacity=cap;

top=-1;


ch=new char[capacity];

}

public void pushChar(char v) {

if(top+1==capacity)

System.out.println("WordPile is full");

else

ch[++top]=v; }

public char popChar() {

if(top== -1) return '\\'; else return ch[top--];

}

public static void main(String args[]) {
```



```

Scanner sc=new Scanner(System.in);

System.out.println("Enter capacity");

int n=sc.nextInt();

if(n>20)
    n=20;

WordPile obj=new WordPile(n);

System.out.println("1. PushCharacter");
System.out.println("2. Pop Character");
System.out.println("3. Display stack");
System.out.println("4. Exit");
while(true){
    System.out.print("Enter your choice:");
    int choice=sc.nextInt();
    char c;
    if(choice==1){
        System.out.print("Enter the character:");
        c=sc.next().charAt(0);
        obj.pushChar(c);
    }if(choice==2){
        c=obj.popChar();
        if(c=='\\') System.out.println("Empty stack.");
        else
            System.out.println(c+" popped.");
    }if(choice==3){
        obj.display();
    }if(choice==4){
        System.exit(0);
    }
}

```

$$\left. \begin{array}{l} \} \\ \} \\ \} \\ \} \end{array} \right\}$$

Variable Description Table

Variable Name	Type	Description
ch	character array	stores the stack of characters
top	int	stores the index of the topmost element
capacity	int	stores the total size of the stack
i	int	loop variable
choice	int	stores the user's choice
n	int	stores the capacity of the stack in main method
W	Object of class WordPile	

Output

```
BlueJ: Terminal Window - TMP
Options
Enter capacity
5
1. PushCharacter
2. Pop Character
3. Display stack
4. Exit
Enter your choice:1
Enter the character:h
Enter your choice:1
Enter the character:e
Enter your choice:1
Enter the character:1
Enter your choice:1
Enter the character:1
Enter your choice:1|
Enter the character:o
Enter your choice:3
    h
    e
    1
    1
    o
Enter your choice:2
o popped.
Enter your choice:3
    h
    e
    1
    1
Enter your choice:2
1 popped.
Enter your choice:2
1 popped.
Enter your choice:2
e popped.
Enter your choice:2
h popped.
Enter your choice:3
Enter your choice:2
Empty stack.
Enter your choice:4
```

Q24) Queue is an entity which can hold a maximum of 100 integers. The queue enables the user to add integers from the rear and remove integers from the front. Define a class Queue with the following details:

Class name : Queue

Data Members / instance variables:

Que[] : array to hold the integer elements

size : stores the size of the array

front : to point the index of the front

rear : to point the index of the rear

Member functions:

Queue (int mm) constructor to initialize the data

size = mm, front = 0, rear = 0

void addele(int v) : to add integer from the rear if possible

else display the message "Overflow"

int delete() : returns elements from front if present, otherwise displays the message "Underflow" and return -9999

void display () : displays the array elements

Specify the class Queue giving details of the functions void addele(int) and int delete() and write the main() method also.

Algorithm

1. START

2. Declare a class named Queue.

a) Initialize integer variables size, front, rear.

b) Initialize an integer array named que.

c) Declare a parametrised constructor and inside check if the size is greater than 100, if not declare size, que, and set front and rear both to 0, else exit showing "Out of range!".

d) Declare a method named addele with parameter int v. Inside check if rear < (size-1).

i) If true, set que[rear]=v, update rear as rear++, print that the element has been added to the queue.

ii) Else, print "Queue overflow".

e) Declare a method named delele and check if front is equal to rear.

i) If true, return -9999.

ii) a) Initialize and declare int removed = que[front].

b) Run a loop through 0 to rear-1, inside setting que[i]=que[i+1].

c) Decrement rear.

d) return removed.

f) Declare a method named show. Check if front is equal to rear.

i) If true, print "Empty"

ii) Else, run a loop from front to rear-1, incrementing the variable. Inside print que[i].

3. Declare a main method and inside declare variable s=100. Call the methods accordingly.

4. END

Source Code

```
import java.util.*;
public class Queue
{
    int que[];
    int size,front,rear;
    public Queue(int mm)
    {
        if(mm>100){
            System.out.println("Out of range!");
            System.exit(0);
        }
        size=mm;
        que=new int[size];
        front=0;rear=0;
    }
    public void addele(int v)
    {
        if(rear<size-1)
        {
            que[rear]=v;
            rear++;
            System.out.println("Element added to the queue");
        }
        else
        {
            System.out.println("Queue overflow");
        }
    }
    public int delele()
    {
        if(front==rear)
            return (-9999);
        else
        {
            int removed=que[front];
            for (int i=0;i<rear-1 ;i++ ) {
                que[i]=que[i+1];
            }
            rear--;
            return removed;
        }
    }
    public void show()
    {

```

```

        if(front==rear)
            System.out.print("Empty");
        else
        {
            for (int i=front;i<=(rear-1);i++)
                System.out.print(que[i]+" ");
        }
        System.out.println();
    }

    public static void main(String args[])
    {
        int s=100,n=0;
        Scanner sc=new Scanner(System.in);

        System.out.println("Enter capacity:");
        Queue obj=new Queue(sc.nextInt()+1);

        System.out.println("\n\nEnter 1 to add an element to a queue");
        System.out.println("Enter 2 to delete an element from a queue");
        System.out.println("Enter 3 to show elements of a queue");
        System.out.println("Enter any other number to exit");
        System.out.println("Enter your choice");

        while(true){
            switch (sc.nextInt())
            {
                case 1:
                    n=sc.nextInt();
                    obj.addele(n);
                    break;
                case 2:
                    if(n==--9999)
                        System.out.println("Queue underflow");
                    else
                        System.out.println("Element deleted is: "+(obj.delele()));
                    break;
                case 3:
                    System.out.println("Queue status");
                    obj.show();
                    break;
                default:
                    System.exit(0);
            }
        }
    }
}

```

}

Variable Description Table

Variable Name	Type	Description
que	int array	Stores the elements
size	int	Stores the size of the queue
front	int	Stores the index of the first element
rear	int	Stores the index of the last element
i	int	loop variable
n	int	Temporary variable for taking user input
s	int	Stores the maximum capacity of the queue

Output

```
BlueJ: Terminal Window - TMP
Options
Enter capacity:
5

Enter 1 to add an element to a queue
Enter 2 to delete an element from a queue
Enter 3 to show elements of a queue
Enter any other number to exit
Enter your choice
1
23
Element added to the queue
1
34
Element added to the queue
1
45
Element added to the queue
1
56
Element added to the queue
3
Queue status
23    34    45    56

1
67
Element added to the queue
1
49

Queue overflow
3
Queue status
23    34    45    56    67

2
Element deleted is: 23

2
Element deleted is: 34

3
Queue status
45    56    67

2
Element deleted is: 45

2
Element deleted is: 56

3
Queue status
67

2
Element deleted is: 67

2
Element deleted is: -9999
```

Q25) A linear data structure enables the user to add address from rear and remove address from front. Define a class Diary:

Data Members:

Q[]: array to store addresses

size: stores max capacity

start: points the index of the front

end: points the index of the rear

Member functions:

Diary(int max): Constructor to initialize data members.

void pushadd(String n): To add address in the diary from the rear end if possible, else prints "NO SPACE"

String popadd(): To remove and return the address from the front end of the diary if any, else returns "????"

void show(): Displays all the addresses in the diary

Specify the class Diary giving the details of the constructors, void pushadd(), String popadd() and void Display(). Define the main() method to create an object and call the functions accordingly to enable the task.

Algorithm

1. START

2. Declare a class named Diary.

a) Initialize integer variables size, start, end.

b) Initialize an String array named Q.

c) Declare a parametrised constructor and inside declare size,Q, and set start and end both to 0, else exit showing "Out of range!".

d) Declare a method named pushadd with parameter String n. Inside check if end<(size- 1).

i) If true, set Q[end]=n, update end as end++, print that the element has been added to the Queue.

ii) Else, print "NO SPACE".

e) Declare a method named popadd and check if start is equal to end.

i) If true, return "????".

ii) Else, a) Initialize and declare String removed = Q[start].

b) Run a loop through 0 to end-1, inside setting Q[i]=Q[i+1].

c) Decrement end.

d) return removed.

f) Declare a method named show. Check if start is equal to end.

i) If true, print "Empty"

ii) Else, run a loop from start to end-1, incrementing the variable. Inside print Q[i].

3. Declare a main method and inside declare variable s=100. Call the methods accordingly.

4. END

Source Code

```
import java.util.*;
public class Diary{
String Q[];
int size,start,end;
Diary(int max){
size=max;
Q=new String[size];
start=end=0;
}
void pushadd(String n){
if(end<size-1){
Q[end]=n;
end++;
}else{
System.out.println("NO SPACE");
}
}
String popadd(){
if(start==end)
return "????";
else {
String removed=Q[start];
for (int i = 0; i < end - 1; i++) {
Q[i] = Q[i + 1];
}
end--;
return removed;
}
}
void show(){
if(start==end)
System.out.println("Empty");
else{
System.out.println("Queue status: ");
for(int i=start;i<=(end-1);i++){
System.out.print(Q[i]+"\\t");
}
System.out.println();
}
}

public static void main(String args[]){
Scanner nrt=new Scanner(System.in);
System.out.println("Enter capacity:");
Diary D=new Diary(nrt.nextInt()+1);
System.out.println("\\n\\nEnter 1 to add an element to a queue");
```



```

        System.out.println("Enter 2 to delete an element from a queue");
        System.out.println("Enter 3 to show elements of a queue");
        System.out.println("Enter any other number to exit");
        System.out.println("Enter your choice: ");

while(true){
switch(nrt.nextInt()){
case 1:
D.pushadd(nrt.nextLine());
break;
case 2:
System.out.println("Address removed is: "+D.popadd());
break;
case 3:
D.show();
break;
default:
System.exit(0);
}
}
}
}
}

```

Variable Description Table

Variable	Type	Description
Q	String array	Holds the addresses
size	int	stores the max capacity of the array
start	int	stores the position of front of the array
end	int	stores the position of rear of the array
i	int	loop variable

Output

```
BlueJ: Terminal Window - TMP
Options
Enter capacity:
4

Enter 1 to add an element to a queue
Enter 2 to delete an element from a queue
Enter 3 to show elements of a queue
Enter any other number to exit
Enter your choice:
1 string 1
1 string 2
1 string 3
1 hello
1 string5
NO SPACE
3
Queue status:
string 1      string 2      string 3      hello
2
Address removed is: string 1
2
Address removed is: string 2
3
Queue status:
string 3      hello
2
Address removed is: string 3
2
Address removed is: hello
```

```
.....
3
Empty
2
Address removed is: ????
```

Q26) 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

1. START
2. Declare a class named Cartons
 - i) Declare a class variable N to store the total no. of boxes.
3. Declare a method named take_input
 - i) Initialise a Scanner object and take user input for the total no. of boxes.
 - ii) If N is greater than 1000, exit the program
4. Declare a method break_up
 - i) Declare integer variables count=0 and copy=N.

- ii) Run a for loop from 48, decrementing to half its original value till it is greater than or equal to six.
 - iii) Inside the loop, set count as count +(copy/i)
 - iv) Print the carton size used and how many were used
 - v) Update copy as copy%i
 - vi) Outside the loop print the remaining no. of boxes
 - vii) Return count if there are no remaining boxes, else return count+1
5. Declare a main method and create an class object and call methods accordingly.
6. END

Source Code

```
import java.util.*;
class Cartons{
int N;
void take_input(){
Scanner nrt=new Scanner(System.in);
System.out.print("Enter the number of boxes: ");
N=nrt.nextInt();
if(N>1000){
System.out.println("INVALID INPUT");
System.exit(0);
}
}
int break_up(){
int count=0,copy=N;
for(int i=48;i>=6;i=i/2){
count += (copy/i);
System.out.println(i+" x "+(copy/i)+" = "+(i*(copy/i)));
copy=(copy%i);

}
System.out.println("\nRemaining boxes = "+copy);

return (copy>0)?(count+1):(count);
}
public static void main(String args[]){
Cartons C=new Cartons();
C.take_input();
System.out.println("\nTotal number of boxes: "+C.N+"\nTotal cartons needed: "+C.break_up());
}
}
```

Variable Description Table

Variable	Type	Description
N	int	Stores number of boxes

count	int	Stores the number of cartons required
copy	int	Stores a copy of N
i	int	loop variable

Output

```
BlueJ: Terminal Window - TMP
Options
Enter the number of boxes: 726
48 x 15 = 720
24 x 0 = 0
12 x 0 = 0
6 x 1 = 6

Remaining boxes = 0

Total number of boxes: 726
Total cartons needed: 16
Enter the number of boxes: 140
48 x 2 = 96
24 x 1 = 24
12 x 1 = 12
6 x 1 = 6

Remaining boxes = 2

Total number of boxes: 140
Total cartons needed: 6
Enter the number of boxes: 4296
INVALID INPUT
```

Q27) 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 the participant(s) having the highest score.

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

1. START

2. Declare a class named Quiz

i) Initialize three arrays as char arr[5], int marks[], key[].

ii) arr and key are of type char and marks is of type int.

iii) Initialize int N to store the total number of participants.

3. Declare a parameterized constructor taking int n as parameter.

i) Check if n is greater than 3 and less than 11, else exit the program.

ii) Set N=n, declare the arrays as arr= new char[N][5], marks=new int[N], key=new char[5]

4. Declare a method named take_input.

i) Take user input for the participants answer by running a loop from 0 to N-1 and an inner loop from 0 to 4.

ii) Take input for the answer key by running a loop from 0 to 4.

5. Declare a method named display.

i) Display the participants options along with the key by running the same loop as in 4)i and 4)ii.

6. Declare a method named check.

i) Declare a int variable count.

ii) Run a loop as 4)i and 4)ii.

iii) Inside check if key at position j is same as arr at position i,j. If same, increment count.

iv) Outside the inner loop, set marks at position i = count and reset count=0.

v) Print the participants scores

7. Declare a method named get_high.

i) Declare a integer variable max=marks[0].

ii) Run a for loop from 1 to N.

iii) Inside check if marks at position i is greater than max.

iv) If greater, then set max=marks[i] and this way get the highest value.

v) Run a for loop from 0 to N-1 and check which participant(s) have the highest score and print their scores.

8. Declare a main method and call methods accordingly.

9.END

Source Code

```
import java.util.*;
public class Quiz{
char arr[][];
int marks[];
char key[];
int N;
Quiz(int n){
if(n<=3 || n>=11){
System.out.println("INPUT SIZE OUT OF RANGE");
System.exit(0);
}
N=n;
arr= new char[N][5];
marks=new int[N];
key=new char[5];
}

    void take_input(){
System.out.println("Enter the participants answers: ");
Scanner nrt=new Scanner(System.in);
for(int i=0;i<N;i++){
System.out.println("Participant: "+(i+1));
String user_input[]=((nrt.nextLine()).trim()).split(" ");
for(int j=0;j<user_input.length;j++){
arr[i][j]=user_input[j].charAt(0);
}
}

        //taking key input
System.out.println("Enter the key: ");
String user_input2[]=((nrt.nextLine()).trim()).split(" ");
for(int i=0;i<5;i++){
key[i]=user_input2[i].charAt(0);
}
}

    void display(){
for(int i=0; i<N;i++){
System.out.print("Participant: "+(i+1));
for(int j=0;j<5;j++){
System.out.print("\t"+arr[i][j]);
}
System.out.println();
}
}
```



```

        System.out.print("\nKEY: \t");
for(int i=0;i<5;i++){
    System.out.print("\t"+key[i]);
}
System.out.println();
}

    void check(){
//checking scores
        int count=0;
for(int i=0;i<N;i++){
    for(int j=0;j<5;j++){
        if(key[j]==arr[i][j])
            count++;
    }
    marks[i]=count;
    count=0;
    System.out.println("Score of Participant "+(i+1)+" is : "+marks[i]);
}
}

    void get_high(){
int max=marks[0];
for(int i=1;i <N;i++){
    if(marks[i]>max)
        max=marks[i];
}

        System.out.println("Max score: "+max+"\nList of Highest scorer(s)");
for(int i=0;i <N;i++){
    if(marks[i]==max)
        System.out.println("Participant "+(i+1)+" score: "+marks[i]);
}
}

    public static void main(String args[]){
Scanner sc=new Scanner(System.in);
System.out.println("Enter the number of participants: ");
Quiz Q=new Quiz(sc.nextInt());
Q.take_input();
Q.display();
Q.check();
Q.get_high();
}
}

```

Variable Description Table

Variable	Type	Description
arr	Character array	Holds the answers of the participants
marks	int array	Holds the marks of the participants
key	Character array	Holds the correct answers
N	int	Stores the number of participants
i	int	loop variable
count	int	Temporarily counts marks and stores in key
max	int	Stores the max marks

Output

```

BlueJ: Terminal Window - Tmp
Options
Enter the participants answers:
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 the key:
B C D A A
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
Score of Participant 1 is : 0
Score of Participant 2 is : 1
Score of Participant 3 is : 1
Score of Participant 4 is : 1
Score of Participant 5 is : 2
Max score: 2
List of Highest scorer(s)
Participant 5 score: 2

BlueJ: Terminal Window - Tmp
Options
Enter the number of participants:
4
Enter the participants answers:
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
Enter the key:
A C D B B
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
Score of Participant 1 is : 3
Score of Participant 2 is : 1
Score of Participant 3 is : 1
Score of Participant 4 is : 3
Max score: 3
List of Highest scorer(s)
Participant 1 score: 3
Participant 4 score: 3

BlueJ: Terminal Window - Tmp
Options
Enter the number of participants:
12
INPUT SIZE OUT OF RANGE

```

Q28) 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.

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

1. START

2. Declare a class named Cipher

i) Initialize variables as String str, int length.

3. Declare a method named take_input.

i) Take user input for the string.

ii) Check if length is lesser than 3 or greater than 100, if true, exit program

4. Declare a method encrypt

i) Declare String ans="".

ii) Declare a char arrays starting from n and cycling back to m.

example:

```
char letters[]={'n','o','p','q','r','s','t','u','v','w','x','y','z','a','b','c','d','e','f','g','h','i','j','k','l','m'};
```

- iii) Run a for loop from 0 to length-1. Initialize char tmp as tmp=str.charAt(i), where i is the loop variable.
- iv) Inside check if it is a question mark, space, etc, etc. If true, set ans+=tmp.
- v) Check if tmp is upper case, if true, set ans+=((letters[((int)(tmp)-65)]+"").toUpperCase()
- vi) Check if tmp is lower case, if true, set ans+=letters[((int)(tmp)-97)].
- vii) Return the string ans.

5. Declare a main method. Create a new object of the class as Cipher C=new Cipher(). Call methods take_input. Print the encrypted version by printing C.encrypt().

Source Code

```
import java.util.*;
public class Cipher{
String str;
int length;
void take_input(){
Scanner nrt=new Scanner(System.in);
System.out.println("Enter the String: ");
str=(nrt.nextLine()).trim();
length=str.length();
if(length<=3 || length>=100){
System.out.println("INVALID LENGTH");
System.exit(0);
}
}

String encrypt(){
String ans="";
char letters[]={'n','o','p','q','r','s','t','u','v','w','x','y','z','a','b','c','d','e','f','g','h','i','j','k','l','m'};

for(int i=0;i<length;i++){
char tmp=str.charAt(i);

if(tmp=='!' || tmp=='.' || tmp==',' || tmp=='?' || tmp==' '){
ans+=tmp;
}
if(Character.isUpperCase(tmp)){
//alphabets

ans+=((letters[((int)(tmp)-65)]+"").toUpperCase());
}
if(Character.isLowerCase(tmp)){
//letters
ans+=letters[((int)(tmp)-97)];
}
```

```

    }
    }
    return ans;
}
public static void main(String args[]){
    Cipher C=new Cipher();
    C.take_input();
    System.out.println("The cipher text is:\n"+C.encrypt());
}
}

```

Variable Description Table

Variable	Type	Description
str	String	Stores the original sentence.
length	int	Stores the length of str
ans	String	Stores the encrypted string
letters	char array	Stores alphabets starting from n and cycling back to a
tmp	char	Stores character temporarily
i	int	loop variable

Output

```

BlueJ: Terminal Window - Tmp
Options
Enter the String:
Hello! How are you?
The cipher text is:
Uryyb! Ubj ner lbh?
Enter the String:
Encryption helps to secure data.
The cipher text is:
Rapelcgvba urycf gb frpher qngn.
Enter the String:
You
INVALID LENGTH

```