



DON BOSCO SCHOOL, LILUAH
2018-2019

Computer Science Project

NILAVA METYA

CLASS: XII C
ROLL No.: 23
REG. No.: 2006/040

January 14, 2019

Contents

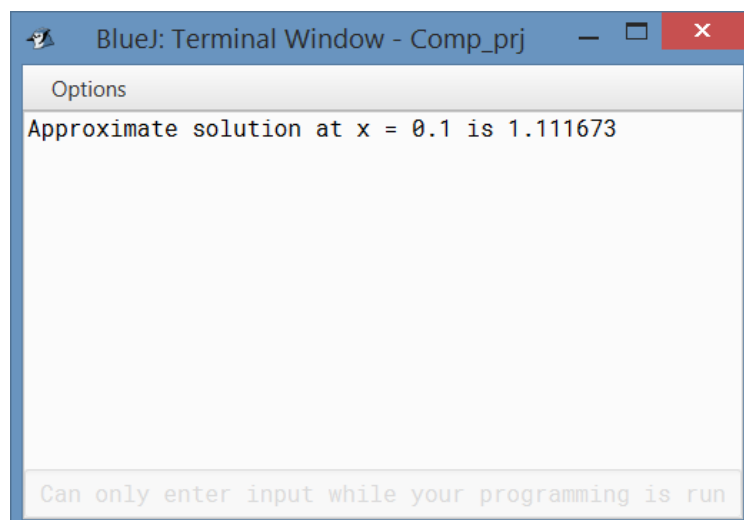
1	Euler's Method	2
2	Fibonacci Numbers	3
3	Angle	5
4	Unique Number	7
5	Partition	11
6	Disarium Number	13
7	Anagrams	15
8	Telephone Directory	17
9	String using Scanner	23
10	Bitwise Sieve	24
11	Hotel Management	26
12	Histogram for frequency of characters	29
13	Employ Salary with interface	31
14	Admission as per Date of birth	33
15	ArrayList demonstration	36
16	StringTokenizer demonstration	37
17	Merge Sort	38
18	Quick Sort	41
19	Traversal in Binary Search Tree	43

1 Euler's Method

Source Code

```
// Program to implement Euler's method for solving differential equations
class Euler
{
    //method to calculate
    float func(float x,float y)
    {
        return (x + y + x*y);
    }
    //method to print solution
    void euler(float x0, float y, float h, float x)
    {
        float temp = 0;
        while(x0 < x)
        {
            temp = y;
            y = y + h*func(x0,y); //approximate solution
            x0 = x0 + h;
        }
        System.out.println("Approximate solution at x = " + x + " is " + y);
    }
    public static void main(String args[])
    {
        Euler obj = new Euler();
        float x0 = 0f;
        float y0 = 1f;
        float h = 0.025f;
        float x = 0.1f;
        obj.euler(x0,y0,h,x); //calling object
    }
}
```

Output

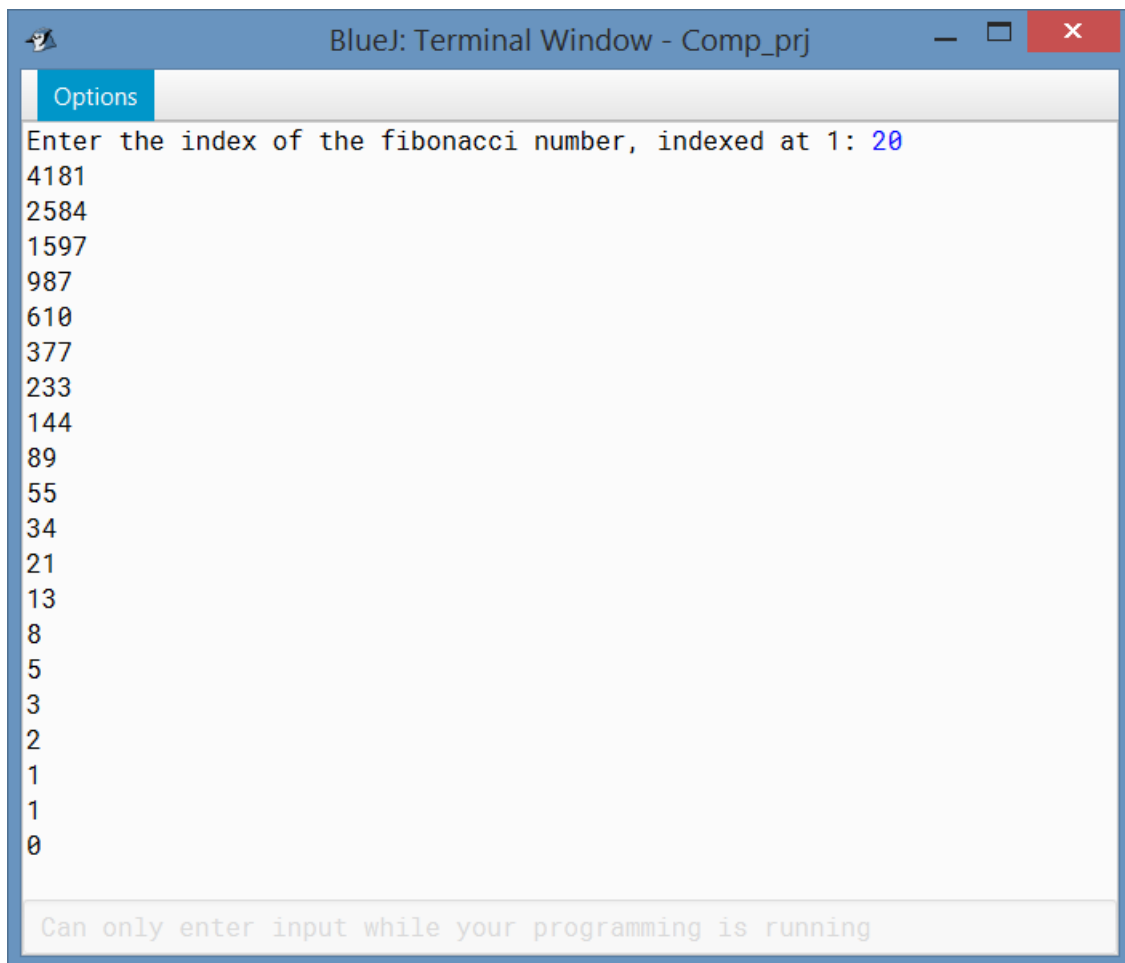


2 Fibonacci Numbers

Source code

```
// Program to print the Fibonacci numbers backwards starting from the nth term
import java.util.*;
class Fibonacci
{
    private long[] f; //array to store the fibonacci numbers
    //method to accept index of last fibonacci number
    public void accept()
    {
        Scanner s = new Scanner(System.in); //Scanner object
        System.out.print("Enter the index of the fibonacci number, indexed at 1: ");
        int n;
        do
        {
            n = s.nextInt();
            if(n < 1) System.out.print("Invalid Input! Please re-enter:");
        } while(n < 1);
        f = new long[n]; //initializing array
        Arrays.fill(f,0); //initializing values of array elements
        f[0] = 0; //base case of Fibonacci sequence
        if(n > 1) f[1] = 1; //base case of Fibonacci sequence
    }
    //recursive function to find Fibonacci numbers
    public long fib(int n)
    {
        if(n == 1) return 1;
        else if(n == 0) return 0;
        else
        {
            if(f[n] == 0) f[n] = fib(n-1) + fib(n-2);
            return f[n];
        }
    }
    //method to print sequence in reverse order
    public void printrev()
    {
        for(int i = f.length-1 ; i >= 0 ; i--)
            System.out.println(fib(i));
    }
    public static void main(String[] args)
    {
        Fibonacci f = new Fibonacci();
        f.accept();
        f.printrev();
    }
}
```

Output



A screenshot of a BlueJ Terminal Window titled "BlueJ: Terminal Window - Comp_prj". The window has a blue title bar with standard window controls (minimize, maximize, close). Below the title bar is a tab labeled "Options". The main area of the terminal is white and contains the following text: "Enter the index of the fibonacci number, indexed at 1: 20" followed by a list of Fibonacci numbers: 4181, 2584, 1597, 987, 610, 377, 233, 144, 89, 55, 34, 21, 13, 8, 5, 3, 2, 1, 1, 0. The number 20 is highlighted in blue. At the bottom of the terminal is a grey bar with the text "Can only enter input while your programming is running".

```
BlueJ: Terminal Window - Comp_prj
Options
Enter the index of the fibonacci number, indexed at 1: 20
4181
2584
1597
987
610
377
233
144
89
55
34
21
13
8
5
3
2
1
1
0
Can only enter input while your programming is running
```

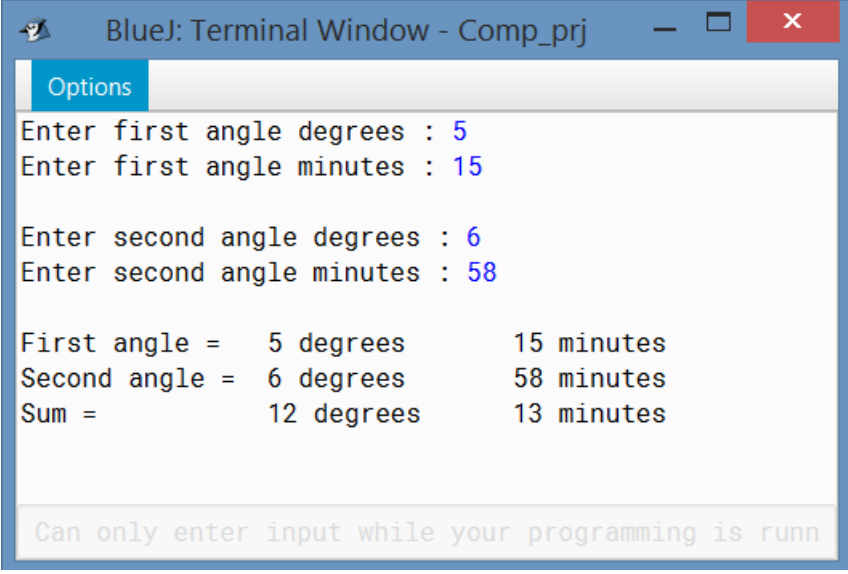
3 Angle

Source Code

```
// Program to implement angle summation
import java.util.*;
class Angle
{
    private int degrees, minutes; //data members
    //method to find sum of two angles
    public static Angle sumangle(Angle a, Angle b)
    {
        Angle ob = new Angle();
        ob.degrees = a.degrees + b.degrees;
        ob.minutes = a.minutes + b.minutes;
        if(ob.minutes >= 60) //degree is always less than 60
        {
            ob.minutes = ob.minutes - 60;
            ob.degrees++;
        }
        return ob;
    }
    //non-parameterized constructor
    public Angle()
    {
        this.degrees = 0;
        this.minutes = 0;
    }
    //parameterized constructor
    public Angle(int x,int y)
    {
        this.degrees = x;
        this.minutes = y;
        while(this.minutes >= 60)
        {
            this.minutes = this.minutes - 60;
            this.degrees++;
        }
    }
    public static void main(String[] args)
    {
        Scanner sc = new Scanner(System.in);
        Angle a = new Angle();
        Angle b = new Angle();
        System.out.print("Enter first angle degrees : ");
        a.degrees = sc.nextInt();
        System.out.print("Enter first angle minutes : ");
        a.minutes = sc.nextInt();
        System.out.print("\nEnter second angle degrees : ");
        b.degrees = sc.nextInt();
        System.out.print("Enter second angle minutes : ");
        b.minutes = sc.nextInt();
        Angle c = sumangle(a,b);
    }
}
```

```
System.out.print("\nFirst angle =\t" + a.degrees + " degrees\t" + a.minutes  
    + " minutes");  
System.out.print("\nSecond angle =\t" + b.degrees + " degrees\t" + b.minutes  
    + " minutes");  
System.out.print("\nSum =\t\t" + c.degrees + " degrees\t" + c.minutes + "  
    minutes");  
}  
}
```

Output



```
BlueJ: Terminal Window - Comp_prj  
Options  
Enter first angle degrees : 5  
Enter first angle minutes : 15  
  
Enter second angle degrees : 6  
Enter second angle minutes : 58  
  
First angle =    5 degrees      15 minutes  
Second angle =    6 degrees      58 minutes  
Sum =           12 degrees      13 minutes  
  
Can only enter input while your programming is running
```

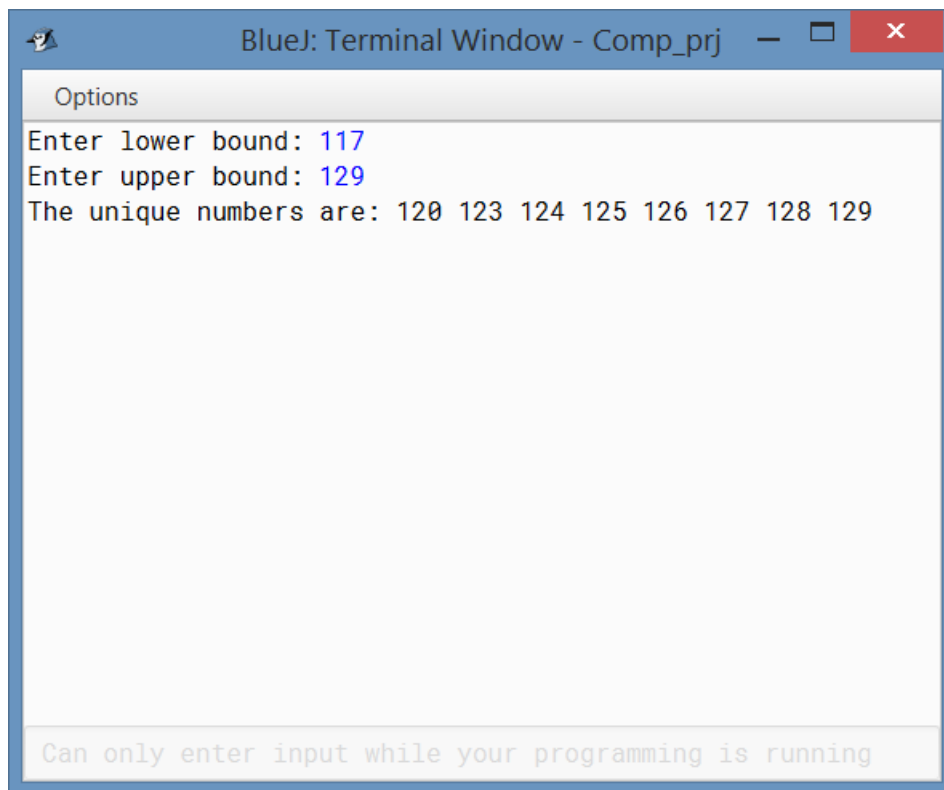
4 Unique Number

4.1 Using String

Source Code

```
// Program to print all Unique numbers within a given range
import java.util.*;
class UniqueNumber
{
    private int m, n;
    //non-parameterized constructor
    public UniqueNumber()
    {
        m = n = 0;
    }
    //function to accept upper and lower limits
    public void accept()
    {
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter lower bound: ");
        m = sc.nextInt();
        System.out.print("Enter upper bound: ");
        n = sc.nextInt();
    }
    //function to check if n has all unique digits
    public boolean check(int n)
    {
        int[] d = new int[10]; //array to store frequency of digits
        Arrays.fill(d,0);
        String s = Integer.toString(n); //converting the number into String
        int l=s.length();
        for(int i=0;i<l;i++) //loop to check for repeated digits
        {
            int dig = Integer.parseInt(s.charAt(i) + "");
            if(d[dig] == 1) return false;
            else d[dig]++;
        }
        return true;
    }
    public static void main(String[] args)
    {
        UniqueNumber u = new UniqueNumber();
        u.accept();
        System.out.print("The unique numbers are: ");
        for(int i= u.m ; i <= u.n ; i++) System.out.print((u.check(i))? (i+" "): "");
    }
}
```

Output



A screenshot of a BlueJ Terminal Window titled "BlueJ: Terminal Window - Comp_prj". The window has a blue title bar with standard window controls. Below the title bar is a tab labeled "Options". The main area of the terminal displays the following text: "Enter lower bound: 117", "Enter upper bound: 129", and "The unique numbers are: 120 123 124 125 126 127 128 129". At the bottom of the window, a light gray status bar contains the text "Can only enter input while your programming is running".

```
Options
Enter lower bound: 117
Enter upper bound: 129
The unique numbers are: 120 123 124 125 126 127 128 129

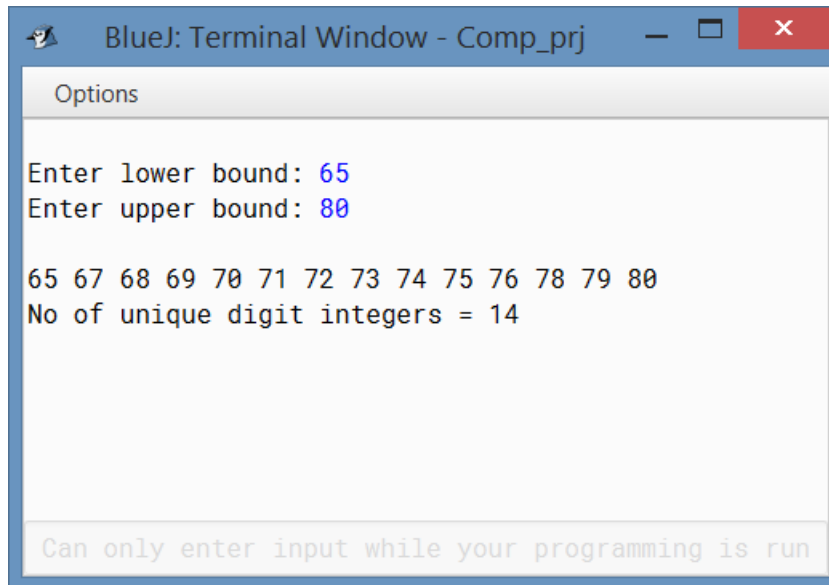
Can only enter input while your programming is running
```

4.2 Using boolean

Source Code

```
//Program to check if all digits are unique unsing Boolean values
import java.util.Scanner;
class UniqueNumBool
{
    public static boolean checkUnique(int n)
    {
        boolean check[] = new boolean[10];
        while(n != 0)
        {
            int d = n % 10; //last digit
            if(check[d]) return false; //repeated presence
            else check[d]=true;
            n = n/10; //remove last digit
        }
        return true;
    }
    public static void main(String[] args)
    {
        int m, n, count = 0;
        Scanner s = new Scanner(System.in);
        System.out.print("\nEnter lower bound: ");
        m = s.nextInt();
        System.out.print("Enter upper bound: ");
        n = s.nextInt();
        System.out.println();
        for(int i = m ; i <= n ; i++) //check in given range
            if(checkUnique(i))
            {
                System.out.print(i + " ");
                count++; //count number of such numbers
            }
        System.out.println("\nNo of unique digit integers = " + count);
    }
}
```

Output

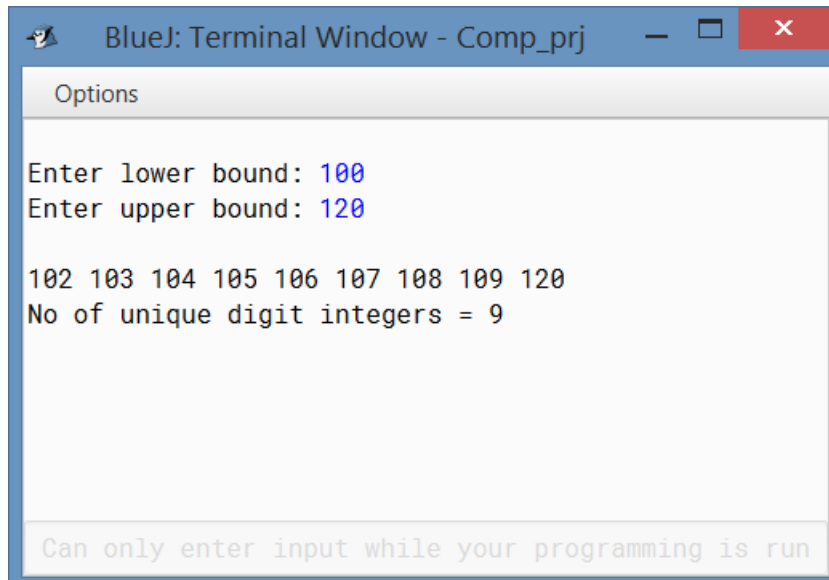


```
Options

Enter lower bound: 65
Enter upper bound: 80

65 67 68 69 70 71 72 73 74 75 76 78 79 80
No of unique digit integers = 14

Can only enter input while your programming is run
```



```
Options

Enter lower bound: 100
Enter upper bound: 120

102 103 104 105 106 107 108 109 120
No of unique digit integers = 9

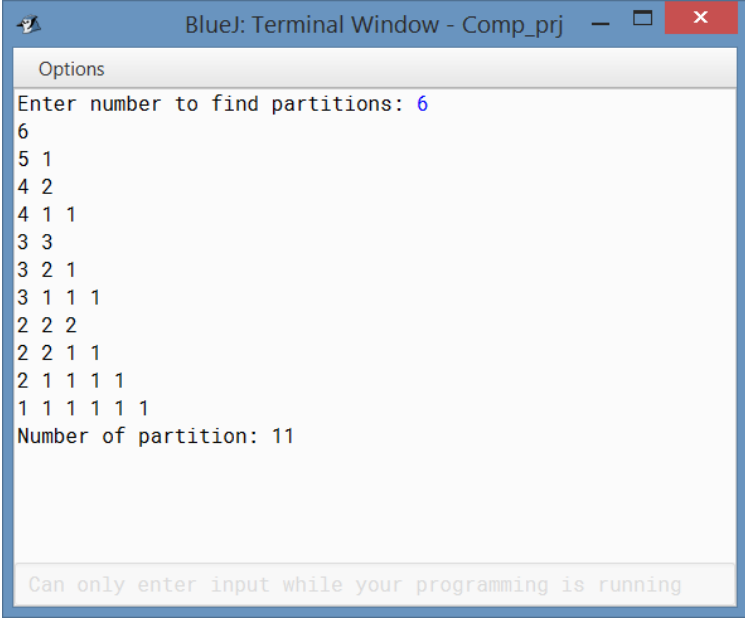
Can only enter input while your programming is run
```

5 Partition

Source Code

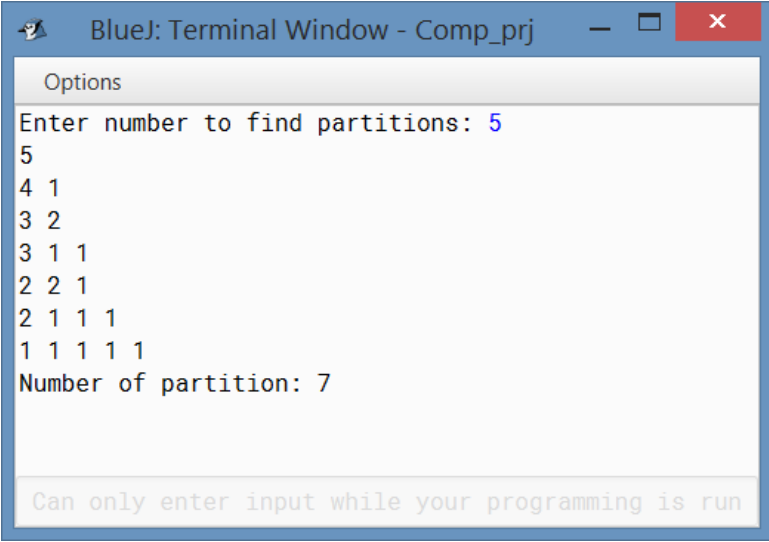
```
// Program to find partitions of a given number
/**
 * Use dynamic programming approach:
 * If  $p(n,k)$  is the number of partitions of  $n$  with highest part at most  $k$ ,
 * then,  $p(n,k) = p(n,k-1) + p(n-k,k)$ 
 * The above formula also helps in establishing a bijection:
 * > If ' $k$ ' is present, then partition the remainder of ' $n-k$ '
 * > If ' $k$ ', not present, then highest part is atmost ' $k-1$ ' (recursive step)
 */
import java.util.*;
class Part
{
    private static int count;
    //recursive function to generate partitions
    public static void partition(int n, int max, String part)
    {
        if(n==0)    //base case
        {
            System.out.println(part);
            count++;
            return;
        }
        for(int i = Math.min(max,n) ; i >= 1 ; i--)
            partition(n-i , i , part + (i + " ")); //recursive step
    }
    public static void main(String[] args)
    {
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter number to find partitions: ");
        int n = sc.nextInt();
        count = 0;
        partition(n,n,"");
        System.out.println("Number of partition: "+count);
    }
}
```

Output



```
Options
Enter number to find partitions: 6
6
5 1
4 2
4 1 1
3 3
3 2 1
3 1 1 1
2 2 2
2 2 1 1
2 1 1 1 1
1 1 1 1 1 1
Number of partition: 11

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



```
Options
Enter number to find partitions: 5
5
4 1
3 2
3 1 1
2 2 1
2 1 1 1
1 1 1 1 1
Number of partition: 7

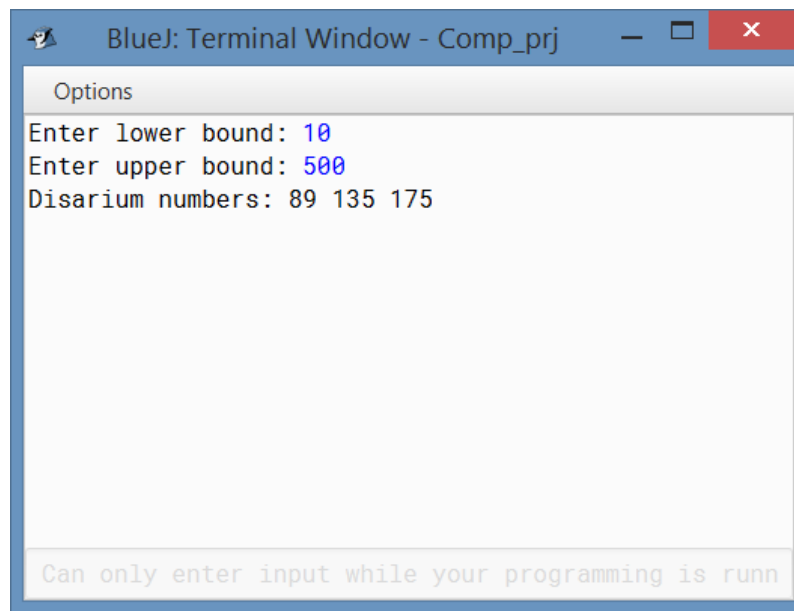
Can only enter input while your programming is run
```

6 Disarium Number

Source Code

```
// Program to find al Disarium numbers in a given range
import java.util.Scanner;
class Disarium
{
    //function to check Disarium number
    public boolean checkDisarium(int num)
    {
        int copy = num, d = 0, sum = 0;
        String s = num + "";
        int len = s.length();
        while(copy > 0)
        {
            d = copy % 10; //last digit
            sum = sum + (int)Math.pow(d,len); //raise to position power
            len--;
            copy = copy / 10; //remove last digit
        }
        if(sum == num) return true;
        else return false;
    }
    public static void main(String[] args)
    {
        Scanner sc = new Scanner(System.in); //Scanner object
        System.out.print("Enter lower bound: ");
        int m=sc.nextInt();
        System.out.print("Enter upper bound: ");
        int n=sc.nextInt();
        Disarium d = new Disarium();
        System.out.print("Disarium numbers:");
        for(int i=m;i<=n;i++)System.out.print(d.checkDisarium(i)? i+" ":"");
    }
}
```

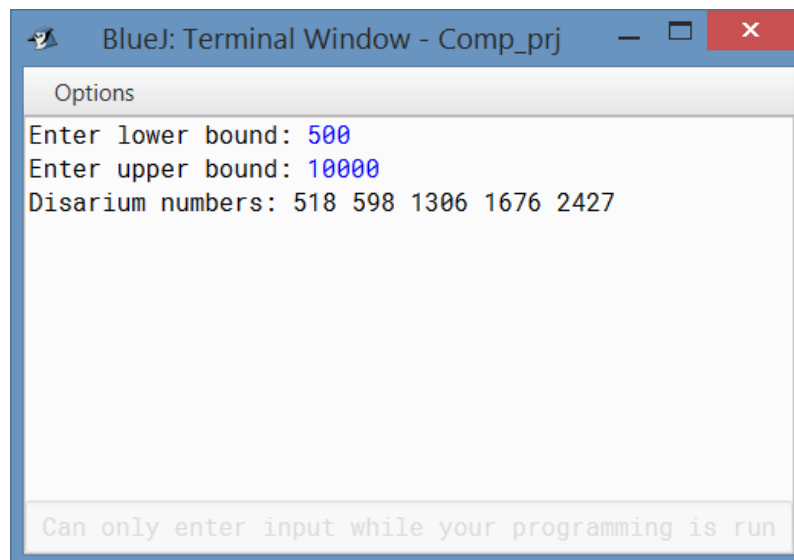
Output



A screenshot of a BlueJ terminal window titled "BlueJ: Terminal Window - Comp_prj". The window has a blue title bar with standard window controls. Below the title bar is a tab labeled "Options". The main text area contains the following text: "Enter lower bound: 10", "Enter upper bound: 500", and "Disarium numbers: 89 135 175". At the bottom of the window, there is a light gray status bar with the text "Can only enter input while your programming is runn".

```
Options
Enter lower bound: 10
Enter upper bound: 500
Disarium numbers: 89 135 175

Can only enter input while your programming is runn
```



A screenshot of a BlueJ terminal window titled "BlueJ: Terminal Window - Comp_prj". The window has a blue title bar with standard window controls. Below the title bar is a tab labeled "Options". The main text area contains the following text: "Enter lower bound: 500", "Enter upper bound: 10000", and "Disarium numbers: 518 598 1306 1676 2427". At the bottom of the window, there is a light gray status bar with the text "Can only enter input while your programming is run".

```
Options
Enter lower bound: 500
Enter upper bound: 10000
Disarium numbers: 518 598 1306 1676 2427

Can only enter input while your programming is run
```

7 Anagrams

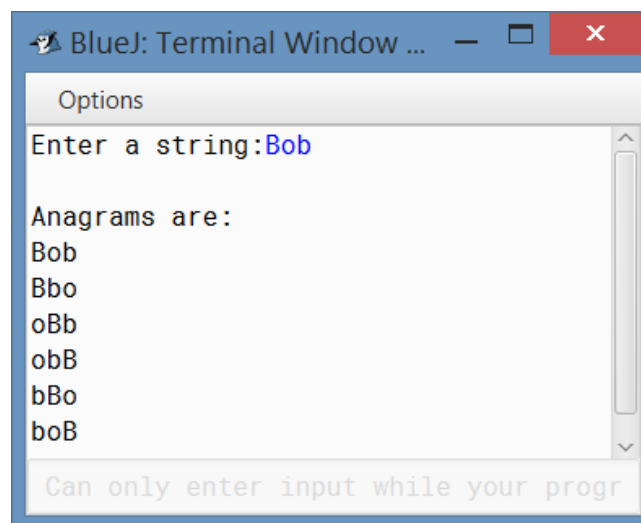
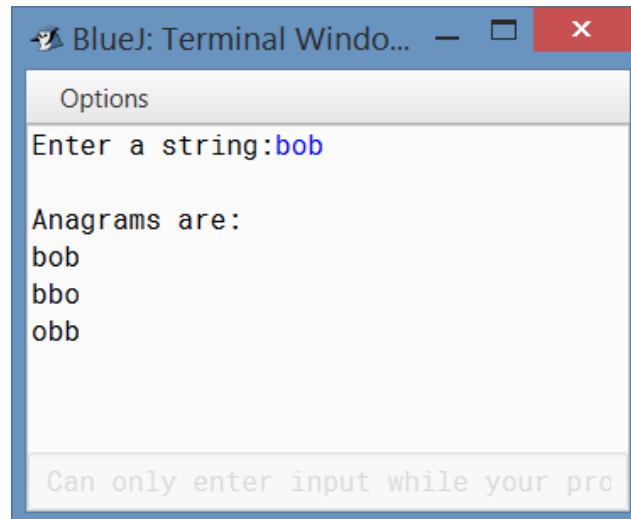
Source Code

```
// Program to find all possible anagrams of a given word
import java.util.*;
class Anagram
{
    private String s;
    //Non-parameterized constructor
    public Anagram()
    {
        s = "";
        accept();
    }
    //method to accept word from user
    public void accept()
    {
        Scanner sc = new Scanner (System.in);
        System.out.print("Enter a string:");
        char[] c = sc.next().toCharArray(); //sorting charactrers of word
        for(int i = 0 ; i < c.length ; i++) s += c[i];
        sc.close();
    }
    //recursive function to find all anagrams
    public void permute(String s, ArrayList<String> a)
    {
        byte[] mark = new byte[256];
        Arrays.fill(mark, (byte)0);
        if(s.length() == 0) a.add(""); //base case
        for(int i = 0 ; i < s.length() ; i++)
        {
            char c = s.charAt(i);
            if(mark[(int) c] == 1) continue; //avoiding repetition
            mark[(int) c] = (byte) 1;
            String r = ""; //string to be permuted
            for(int j = 0 ; j < s.length() ; j++)
            {
                if(j == i) continue; //permute all except current character
                r += s.charAt(j);
            }
            ArrayList<String> temp = new ArrayList<String>();
            permute(r, temp); //recursive step
            for(int j = 0 ; j < temp.size() ; j++) a.add(c + temp.get(j));
        }
    }
    public static void main(String[] args)
    {
        Anagram p = new Anagram(); //Anagram object
        ArrayList<String> a = new ArrayList<String>();
        p.permute(p.s,a);
        System.out.println("\nAnagrams are:");
        for(int i = 0 ; i < a.size() ; i++)
```



```
        System.out.println(a.get(i));  
    }  
}
```

Output



8 Telephone Directory

Source Code

```
//Program to maintain a telephone directory
import java.util.*;
import java.io.*;
class TelDirectory
{
    private ArrayList<Data> dir = new ArrayList<Data>(); //stores all data
    //non-parameterized constructor
    public TelDirectory()throws IOException
    {
        BufferedReader br = new BufferedReader(new FileReader("Directory.txt"));
        String id = "";
        while(id != null)
        {
            id = br.readLine();
            if(id == null) break;
            int Id = Integer.parseInt(id);
            Data d = new Data(Id,br.readLine(),Long.parseLong(br.readLine()));
            dir.add(d);
        }
        br.close();
    }
    //method to add data
    public void append()throws IOException
    {
        System.out.println("Adding");
        int id = dir.size() + 1;
        PrintWriter p = new PrintWriter(new BufferedWriter(new
            FileWriter("Directory.txt",true)));
        BufferedReader br = new BufferedReader(new InputStreamReader(System.in));
        System.out.print("Enter name: ");
        String name = br.readLine();
        System.out.print("Enter phone : ");
        long phone = Long.parseLong(br.readLine());
        dir.add(new Data(id,name,phone));
        p.println(id); //writing in file
        p.println(name);
        p.println(phone);
        p.close();
    }
    //method to delete data
    public void remove()throws IOException
    {
        System.out.println("Deleting");
        System.out.print("Delete by:\n1.Name\t2.Phone\nEnter choice: ");
        BufferedReader br = new BufferedReader(new InputStreamReader(System.in));
        char choice = br.readLine().charAt(0);
        if(choice == '1') //search by name
        {
            System.out.print("Enter name: ");
        }
    }
}
```

```

        String name = br.readLine();
        Data d = search(name);
        if(d != null) dir.remove(d.id);
        else System.out.println("Name not found.");
    }
    else if(choice == '2') //search by phone
    {
        System.out.print("Enter phone: ");
        long phone = Long.parseLong(br.readLine());
        Data d = search(phone);
        if(d!=null) dir.remove(d.id);
        else System.out.println("Phone not found.");
    }
    PrintWriter p = new PrintWriter(new BufferedWriter(new
        FileWriter("Directory.txt")));
    for(int i = 0 ; i < dir.size() ; i++)
        p.println(i+"\n"+dir.get(i).name+"\n"+dir.get(i).phone);
    p.close();
}
//method to edit contents of directory
public void edit() throws IOException
{
    System.out.println("Updating");
    System.out.println("Search by:\n1.Name\t2.Phone\n Enter choice: ");
    BufferedReader br = new BufferedReader(new InputStreamReader(System.in));
    char choice = br.readLine().charAt(0);
    if(choice == '1') //search by name
    {
        System.out.print("Enter name: ");
        String name = br.readLine();
        Data d = search(name);
        if(d!=null)
        {
            System.out.print("New Name: ");
            d.name = br.readLine();
            System.out.print("New Phone: ");
            d.phone = Long.parseLong(br.readLine());
            dir.set(d.id,d);
        }
        else System.out.println("Name not found.");
    }
    else if(choice == '2') //search by phone
    {
        System.out.print("Enter phone: ");
        long phone = Long.parseLong(br.readLine());
        Data d = search(phone);
        if(d!=null)
        {
            System.out.print("New Name: ");
            d.name = br.readLine();
            System.out.print("New Phone: ");
            d.phone = Long.parseLong(br.readLine());
            dir.set(d.id,d);
        }
    }
}

```

```

        else System.out.println("Phone not found.");
    }
    PrintWriter p = new PrintWriter(new BufferedWriter(new
        FileWriter("Directory.txt")));
    for(int i = 0 ; i < dir.size() ; i++) //update file
    {
        Data d = dir.get(i);
        p.println(i+"\n"+d.name+"\n"+d.phone);
    }
    p.close();
}
//method to search and show results
public void search() throws IOException
{
    System.out.println("Searching");
    System.out.println("Search by:\n1.Name\t2.Phone");
    BufferedReader br = new BufferedReader(new InputStreamReader(System.in));
    char choice = br.readLine().charAt(0);
    if(choice == '1') //search by name
    {
        System.out.print("Enter name: ");
        String name = br.readLine();
        Data d = search(name);
        if(d!=null) System.out.println("Id: " + d.id+"\nName: " +
            d.name+"\nPhone: " + d.phone);
        else System.out.println("Name not found.");
    }
    else if(choice == '2') //search by phone
    {
        System.out.print("Enter phone: ");
        long phone = Long.parseLong(br.readLine());
        Data d = search(phone);
        if(d!=null) System.out.println("Id: " + d.id+"\nName: " +
            d.name+"\nPhone: " + d.phone);
        else System.out.println("Phone not found.");
    }
}
//method to search by name
private Data search(String name)
{
    for(int i = 0 ; i < dir.size() ; i++)
    {
        Data d = dir.get(i);
        if(d.name.equals(name)) return d;
    }
    return null;
}
//method to search by phone
private Data search(long phone)
{
    for(int i = 0 ; i < dir.size() ; i++)
    {
        Data d = dir.get(i);
        if(d.phone==phone) return d;
    }
}

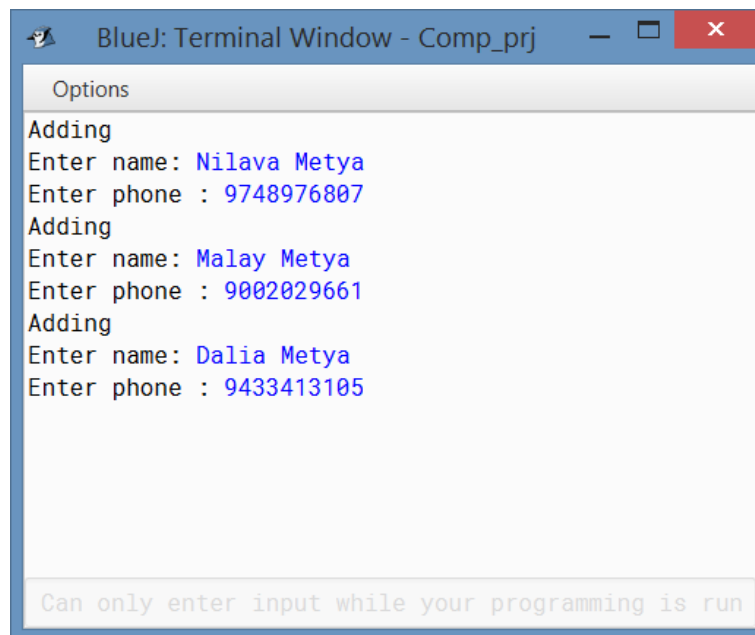
```

```

    }
    return null;
}
//method to print entire list
public void displayWholeList()
{
    System.out.println("Id\tPhone\t\tName");
    System.out.println(".....\n");
    for(int i = 0 ; i < dir.size() ; i++)
    {
        Data d = new Data();
        d = dir.get(i);
        System.out.println(" " + d.id + "\t" + d.phone + "\t" + d.name);
    }
}
}
//user-defined data type
class Data
{
    private int id;
    private String name;
    private long phone;
    //non-parameterized constructor
    public Data()
    {
        id = 0;
        name = "";
        phone = 0;
    }
    //parameterized constructor
    public Data(int x, String y, long z)
    {
        id = x;
        name = y;
        phone = z;
    }
}
}

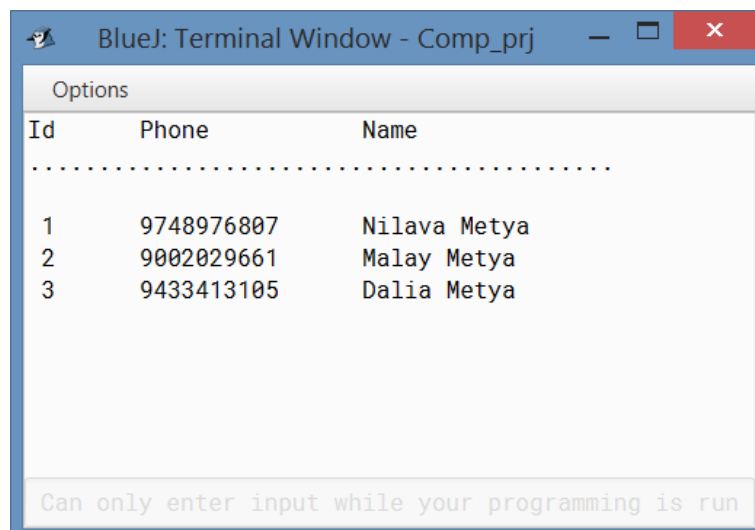
```

Output



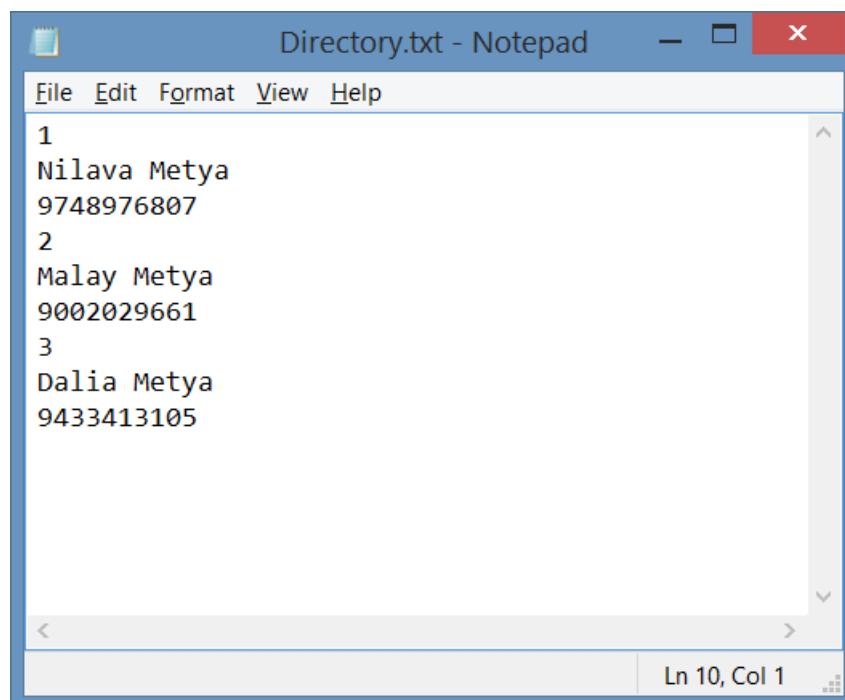
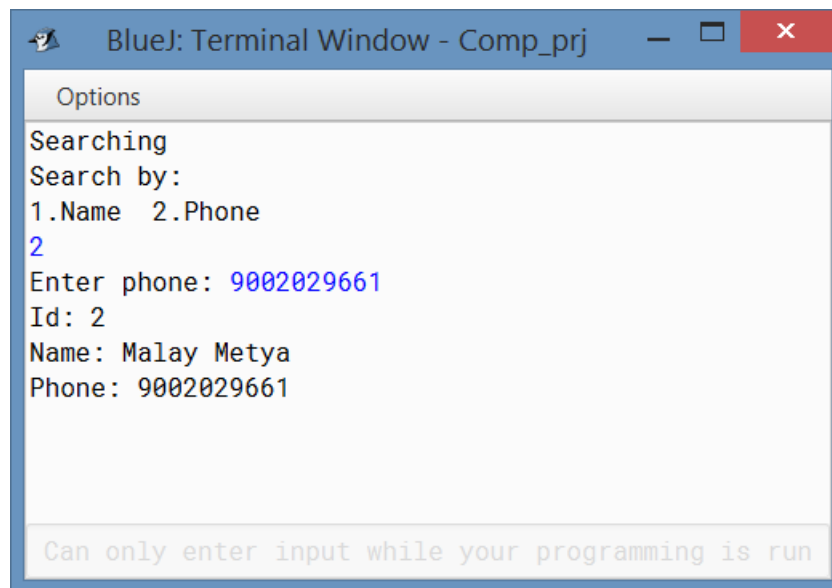
```
BlueJ: Terminal Window - Comp_prj
Options
Adding
Enter name: Nilava Metya
Enter phone : 9748976807
Adding
Enter name: Malay Metya
Enter phone : 9002029661
Adding
Enter name: Dalia Metya
Enter phone : 9433413105

Can only enter input while your programming is run
```



```
BlueJ: Terminal Window - Comp_prj
Options
Id      Phone      Name
.....
1       9748976807   Nilava Metya
2       9002029661   Malay Metya
3       9433413105   Dalia Metya

Can only enter input while your programming is run
```

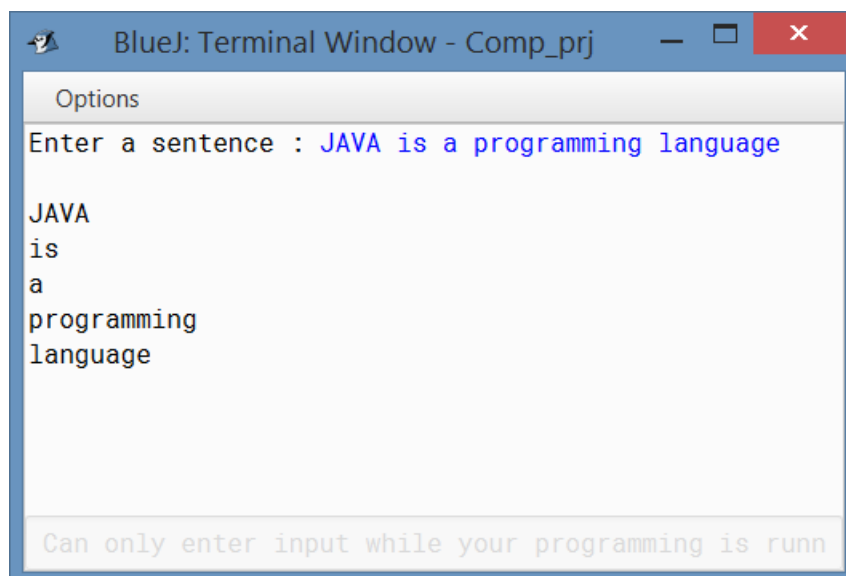


9 String using Scanner

Source Code

```
// Program to find tokens from a string using Scanner
import java.util.*;
import java.io.*;
class StringScanner
{
    public static void main(String args[])
    {
        String s = "";
        System.out.print("Enter a sentence : ");
        try
        {
            BufferedReader br = new BufferedReader(new InputStreamReader(System.in));
            s = br.readLine(); //input string from user
        }
        catch(Exception e) //Exception handling
        {
            System.out.println(e.getMessage());
        }
        System.out.println();
        Scanner sc = new Scanner(s);
        while(sc.hasNext()) //check for more tokens
        {
            System.out.println(sc.next()); //extract tokens
        }
    }
}
```

Output



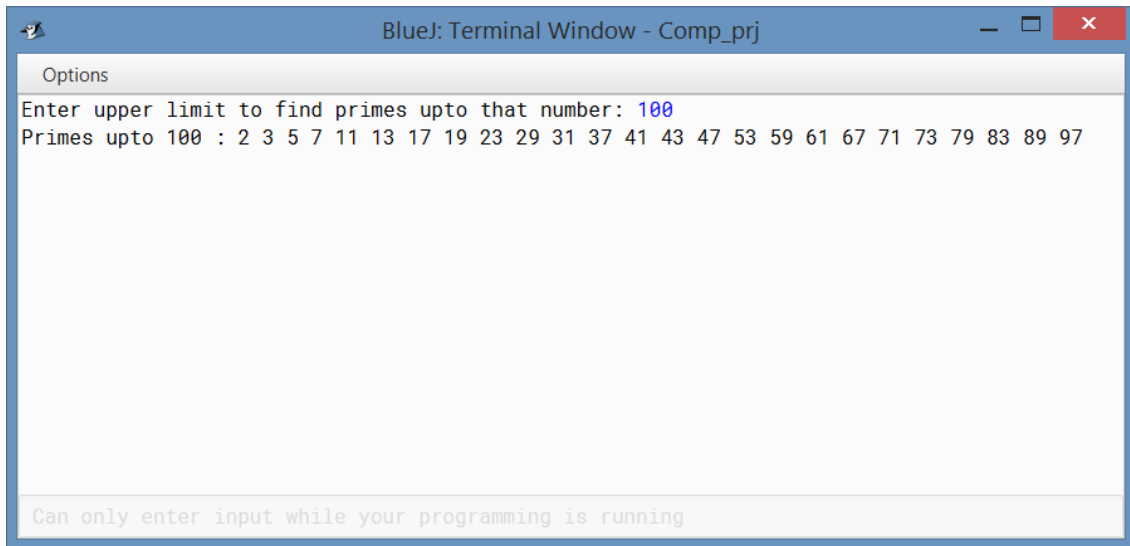
10 Bitwise Sieve

Source Code

```
// Program to implement Bitwise Sieve of Eratosthenes
import java.util.*;
class BitwiseSieve
{
    //method to check whether x is prime or composite
    public static int isComposite(int prime[], int x)
    {
        // checking whether the value of element is set or not
        // Using prime[x/64], find the slot in prime array
        // To find the bit number, divide x by 2 and take its mod with 32
        return (prime[x/64] & (1 << ((x >> 1) & 31)));
    }
    //marks x composite in prime[]
    public static void makeComposite(int prime[], int a)
    {
        // Set a bit corresponding to given element
        // To find the bit number divide a by 2 and take it mod 32
        prime[a / 64] |= (1 << ((a >> 1) & 31));
    }
    //method to print all primes smaller than n
    public static void bitWiseSieve(int n)
    {
        // Assuming that n takes 32 bits, reduce size to n/64 from n/2
        int prime[] = new int[n/64 + 1];
        // 2 is the only even prime so check only odd primes
        for (int i = 3; i * i <= n; i += 2)
        {
            // If i is prime, mark all its multiples as composite
            if (isComposite(prime, i) == 0)
                for (int j = i * i, k = i << 1; j < n; j += k)
                    makeComposite(prime, j);
        }
        System.out.print("Primes upto " + n + " : 2 "); //print 2 separately
        //print other primes
        for (int i = 3; i <= n; i += 2)
            if (isComposite(prime, i) == 0)
                System.out.print(i+" ");
    }
    public static void main(String[] args)
    {
        int n;
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter upper limit to find primes upto that number: ");
        do
        {
            n = sc.nextInt();
            if(n < 1) System.out.print("Invalid input! Re-enter: ");
        } while(n < 1);
        if(n == 1) System.out.print("No primes upto 1!");
    }
}
```

```
        else bitWiseSieve(n);  
    }  
}
```

Output



11 Hotel Management

Source Code

```
// Program for Hotel management
import java.util.Scanner;
class HotelExe
{
    public static void main(String args[])
    {
        HotelManagement hm = new HotelManagement();
        hm.getDetails();
        hm.bookRoom();
    }
}
class HotelManagement
{
    private Scanner sc = new Scanner(System.in);
    private int month, year, roomCount, dayCount, currId;
    private int rooms[][];
    private Details ids[];
    //non-parameterized constructor
    public HotelManagement()
    {
        System.out.print("Enter number of rooms: ");
        roomCount = sc.nextInt();
        System.out.print("Enter month number (1 for January,etc): ");
        month = sc.nextInt();
        if(month < 1 || month > 12) System.out.print("Invalid input.");
        else
        {
            if(month == 2) //need year for leap year check
            {
                System.out.print("Enter year: ");
                year = sc.nextInt();
            }
            if (month == 1 || month == 3 || month == 5 || month == 7 || month == 8 ||
                month == 10 || month == 12)
                dayCount = 31;
            else if (month == 4 || month == 6 || month == 9 || month == 11)
                dayCount = 30;
            else //leap year check
            {
                if(year % 400 == 0 || (year % 100 != 0 && year % 4 == 0)) dayCount =
                    29;
                else dayCount = 28;
            }
            rooms = new int[roomCount][dayCount];
            ids = new Details[(roomCount * dayCount)];
        }
        currId = 1;
    }
    //method to book a room
```

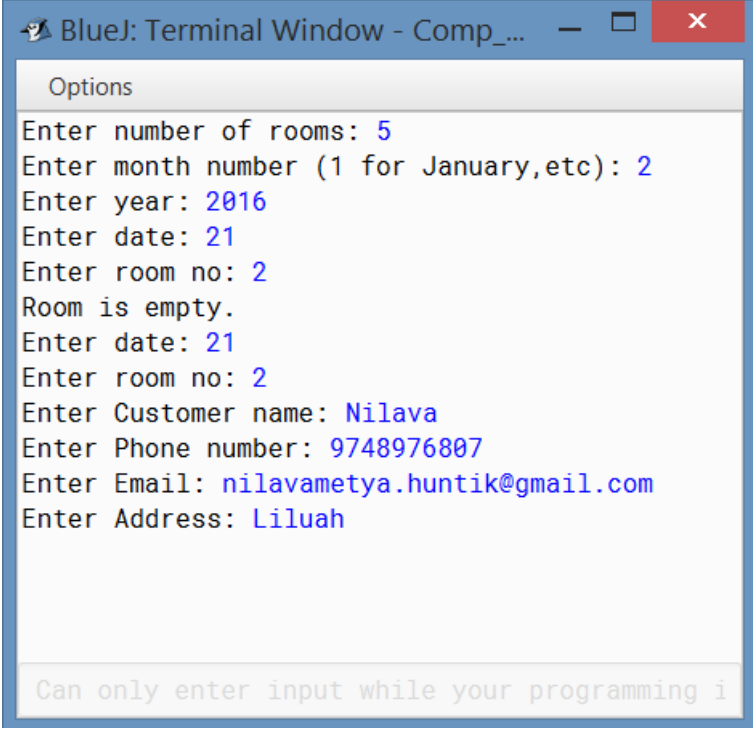
```

void bookRoom()
{
    System.out.print("Enter date: ");
    int date;
    do //check valid date
    {
        date = sc.nextInt();
        if(date < 1 || date > dayCount) System.out.println("Invalid input! Please
            re-enter: ");
    }while(date < 1 || date > dayCount);
    System.out.print("Enter room no: ");
    int roomno = sc.nextInt() - 1; //-1 for array numbering
    int id = rooms[roomno][date];
    if (id == 0) //if id=0, then room is empty
    {
        rooms[roomno][date] = currId;
        Details x = new Details(); //Details object
        System.out.print("Enter Customer name: ");
        x.name = sc.next();
        System.out.print("Enter Phone number: ");
        x.phone = sc.nextLong();
        System.out.print("Enter Email: ");
        x.email = sc.next();
        System.out.print("Enter Address: ");
        x.address = sc.next();
        ids[currId] = x;
        currId++;
    }
    else System.out.println("Room not empty. Try another one.");
}
//method to get details of a customer
void getDetails()
{
    System.out.print("Enter date: ");
    int date;
    do //check valid date
    {
        date = sc.nextInt();
        if(date < 1 || date > dayCount) System.out.println("Invalid input! Please
            re-enter: ");
    } while(date < 1 || date > dayCount);
    System.out.print("Enter room no: ");
    int roomno = sc.nextInt();
    roomno--;
    int id = rooms[roomno][date];
    if(id == 0) System.out.println("Room is empty.");
    else
    {
        Details a = ids[id];
        System.out.println("Customer name: " + a.name + "\nPhone number: " +
            a.phone + "\nEmail: " + a.email + "\nAddress: " + a.address);
    }
}
}
}

```

```
//user defined data type for details
class Details
{
    String name;
    long phone;
    String address;
    String email;
}
```

Output



The screenshot shows a BlueJ Terminal Window titled "BlueJ: Terminal Window - Comp_...". It contains a series of prompts and user inputs. The prompts are in black text, and the user inputs are in blue text. The inputs are: 5, 2, 2016, 21, 2, Nilava, 9748976807, nilavametya.huntik@gmail.com, and Liluah. The window also has an "Options" tab at the top and a status bar at the bottom that says "Can only enter input while your programming i".

```
Options
Enter number of rooms: 5
Enter month number (1 for January,etc): 2
Enter year: 2016
Enter date: 21
Enter room no: 2
Room is empty.
Enter date: 21
Enter room no: 2
Enter Customer name: Nilava
Enter Phone number: 9748976807
Enter Email: nilavametya.huntik@gmail.com
Enter Address: Liluah

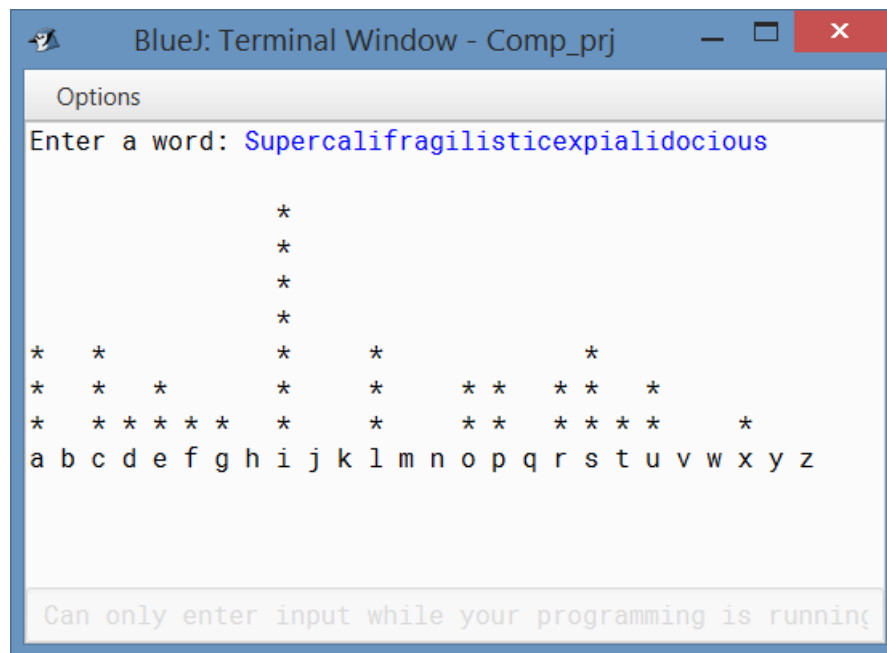
Can only enter input while your programming i
```

12 Histogram for frequency of characters

Source Code

```
// Program to print vertical histogram of frequency of characters
import java.util.*;
class Histogram
{
    private static String s;
    private static int[] a; //array to store frequency of characters
    public static void print()
    {
        System.out.println();
        for(int i = 0 ; i < s.length() ; i++)
        {
            int pos = s.charAt(i)-'a';
            a[pos]++; //calculating character frequency
        }
        int var = 0, max = 0;
        max = a[0];
        for(int i = 0 ; i < a.length ; i++)
            if(a[i] > max) max = a[i];
        var = max;
        for(int i = 0 ; i < var ; i++)
        {
            for(int j = 0 ; j < a.length ; j++)
                if(a[j] < max) System.out.print(" ");
                else System.out.print("* "); //print when character has count at
                    least this level
            max--; //changing level
            System.out.println();
        }
        for(int i = 0 ; i < 26 ; i++)
            System.out.print((char)(i + 'a') + " ");
    }
    public static void main(String[] args)
    {
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter a word: ");
        s = sc.nextLine().toLowerCase();
        a = new int[26];
        Arrays.fill(a,0); //inititalizing character frequencies
        print();
    }
}
```

Output



13 Employ Salary with interface

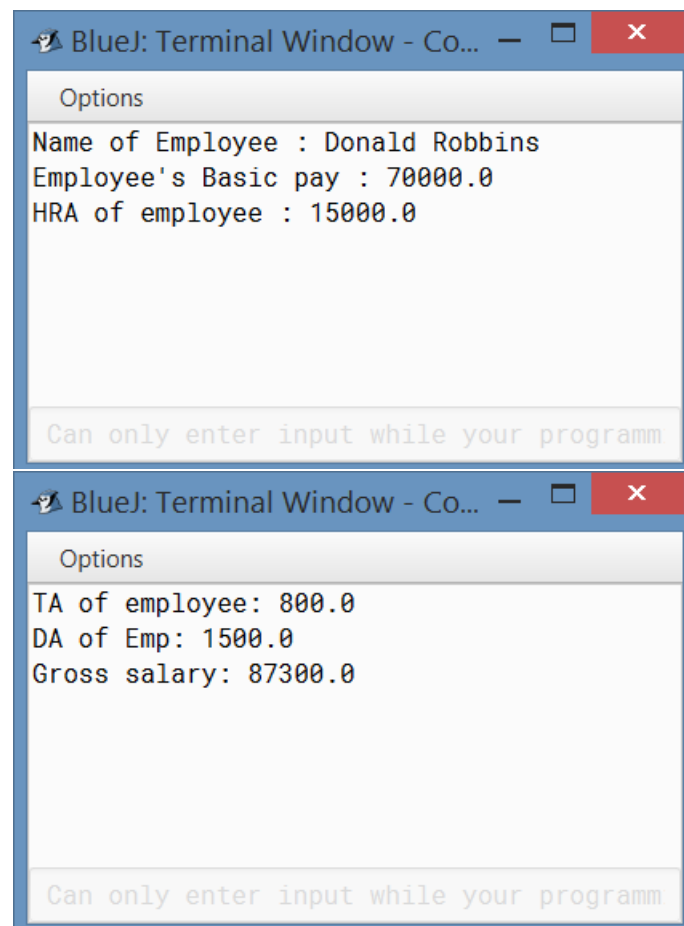
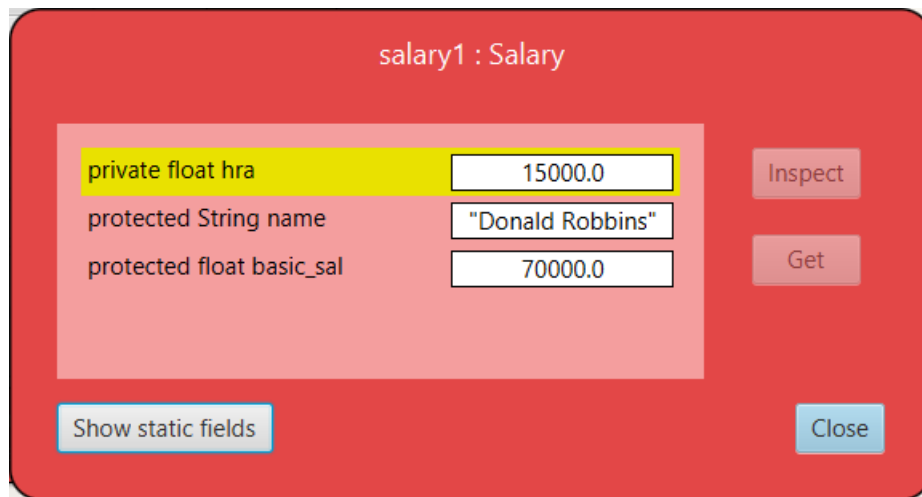
Source Code

```
//Program to implement Employ class using interface
class Salary extends Employee implements gross
{
    private float hra;
    public Salary(String n,float b,float h)
    {
        super(n,b);
        hra = h;
    }
    public void display()
    {
        super.display(); //calling display() from Employee class
        System.out.print("HRA of employee : "+hra);
    }
    public void gross_sal()
    {
        double gross_sal = basic_sal + ta + da + hra; //calculate gross salary
        System.out.println("TA of employee: " + ta);
        System.out.println("DA of Emp: " + da);
        System.out.println("Gross salary: "+ gross_sal);
    }
}
//user defined data type
class EmpDetails
{
    public static void main(String args[])
    {
        Salary s = new Salary("Anirban",8000,8000);
        s.display();
        s.gross_sal();
    }
}
interface gross
{
    double ta = 800.00;
    double da = 1500.00;
    void gross_sal();
}
//user-defined data type
class Employee
{
    protected String name;
    protected float basic_sal;
    //parameterized constructor
    public Employee(String n, float b)
    {
        name = n;
        basic_sal = b;
    }
}
```



```
//method to display details
void display() //default access modifier
{
    System.out.println("Name of Employee : " + name);
    System.out.println("Employee's Basic pay : " + basic_sal);
}
}
```

Output



14 Admission as per Date of birth

Source Code

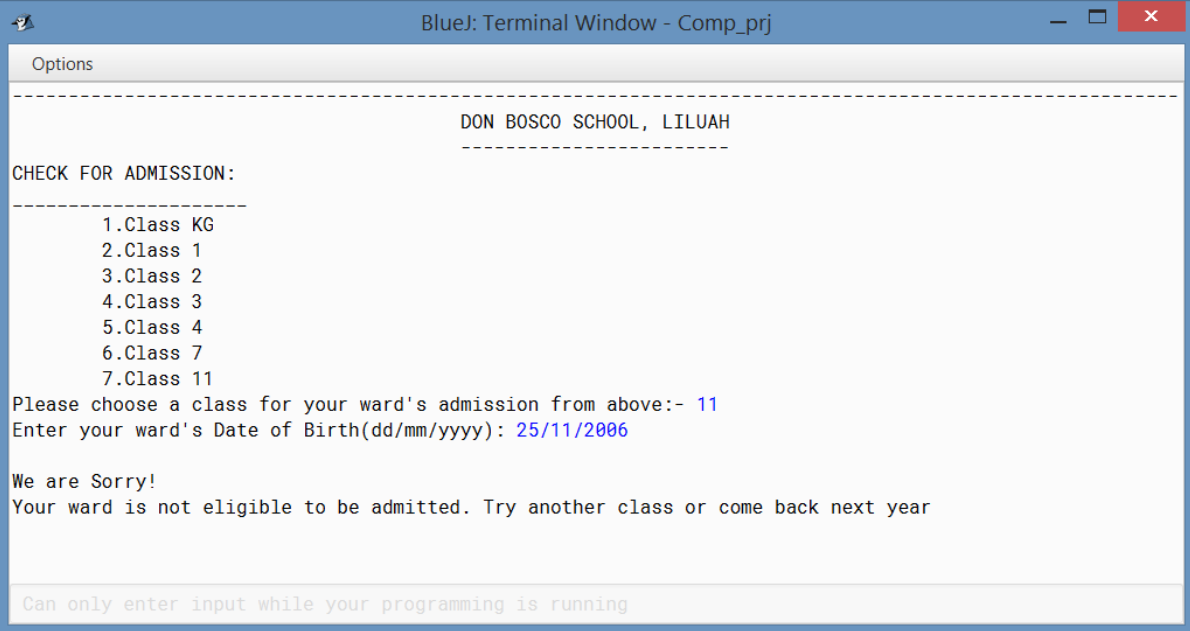
```
//Program to check admission criteria
import java.util.*;
class Admission
{
    private Scanner sc=new Scanner(System.in);
    private int month[]={0,31,28,31,30,31,30,31,31,30,31,30,31};
    //function for checking for Leap Year
    public int isLeap(int y)
    {
        if((y%400==0) || ((y%100!=0)&&(y%4==0))) return 29;
        return 28;
    }
    //function for checking date validation
    public boolean dateValid(int d, int m, int y)
    {
        month[2] = isLeap(y);
        if(m < 0 || m > 12 || d < 0 || d > month[m] || y < 0 || y > 9999)
            return false;
        return true;
    }
    //function for finding day number from year = 1 till the input year
    public int dayno(int d, int m, int y)
    {
        int dn = 0;
        month[2] = isLeap(y);
        for(int i = 1 ; i < m ; i++) dn=dn+month[i];
        dn = dn + d;
        for(int i = 1 ; i < y ; i++)
        {
            if(isLeap(i) == 29) dn = dn + 366;
            else dn = dn + 365;
        }
        return dn;
    }
    public static void main(String[] args)
    {
        Admission aa = new Admission();
        System.out.println("-----");
        System.out.println("\t\t\t\t\tDON BOSCO SCHOOL, LILUAH");
        System.out.println("\t\t\t\t\t-----");
        System.out.println("CHECK FOR ADMISSION: ");
        System.out.println("-----");
        System.out.println("\t1.Class KG");
        System.out.println("\t2.Class 1");
        System.out.println("\t3.Class 2");
        System.out.println("\t4.Class 3");
        System.out.println("\t5.Class 4");
        System.out.println("\t6.Class 7");
        System.out.println("\t7.Class 11");
    }
}
```

```

System.out.print("Please choose a class for your ward's admission from
    above:- ");
int choice = aa.sc.nextInt();
System.out.print("Enter your ward's Date of Birth(dd/mm/yyyy): ");
String date1 = aa.sc.next();
int a, b;
a = date1.indexOf("/"); //Extracting the day
int d1 = Integer.parseInt(date1.substring(0,a));
b = date1.lastIndexOf("/"); //Extracting the month
int m1 = Integer.parseInt(date1.substring(a+1,b));
//Extracting the year
int y1 = Integer.parseInt(date1.substring(b+1));
String date2="31/03/2018";
a = date2.indexOf("/"); //taking '/' for separation
int d2 = Integer.parseInt(date2.substring(0,a));
b = date2.lastIndexOf("/"); //taking '/' for separation
int m2 = Integer.parseInt(date2.substring(a+1,b));
int y2 = Integer.parseInt(date2.substring(b+1));
//Validating both dates
if(aa.dateValid(d1,m1,y1) && aa.dateValid(d2,m2,y2))
{
    int p = aa.dayno(d1,m1,y1);
    int q = aa.dayno(d2,m2,y2);
    double r = Math.abs(p-q);
    boolean flag = false;
    if(choice == 1 && r/365 >= 3 && r/365 < 4) flag=true;
    else if(choice == 2 && r/365 >= 4 && r/365 < 5) flag=true;
    else if(choice == 3 && r/365 >= 5 && r/365 < 6) flag=true;
    else if(choice == 4 && r/365 >= 6 && r/365 < 7) flag=true;
    else if(choice == 5 && r/365 >= 7 && r/365 < 8) flag=true;
    else if(choice == 6 && r/365 >= 9 && r/365 < 10) flag=true;
    else if(choice == 7 && r/365 >= 16 && r/365 < 17) flag=true;
    if(flag) System.out.println("\nCongratulations!\nYour ward is eligible to
        be admitted\nPlease download the form from school website and submit
        at the school reception by 20th April, 2018\n");
    else System.out.println("\nWe are Sorry!\nYour ward is not eligible to be
        admitted. Try another class or come back next year");
}
}
}

```

Output



```
BlueJ: Terminal Window - Comp_prj
Options
-----
DON BOSCO SCHOOL, LILUAH
-----
CHECK FOR ADMISSION:
-----
1.Class KG
2.Class 1
3.Class 2
4.Class 3
5.Class 4
6.Class 7
7.Class 11
Please choose a class for your ward's admission from above:- 11
Enter your ward's Date of Birth(dd/mm/yyyy): 25/11/2006

We are Sorry!
Your ward is not eligible to be admitted. Try another class or come back next year

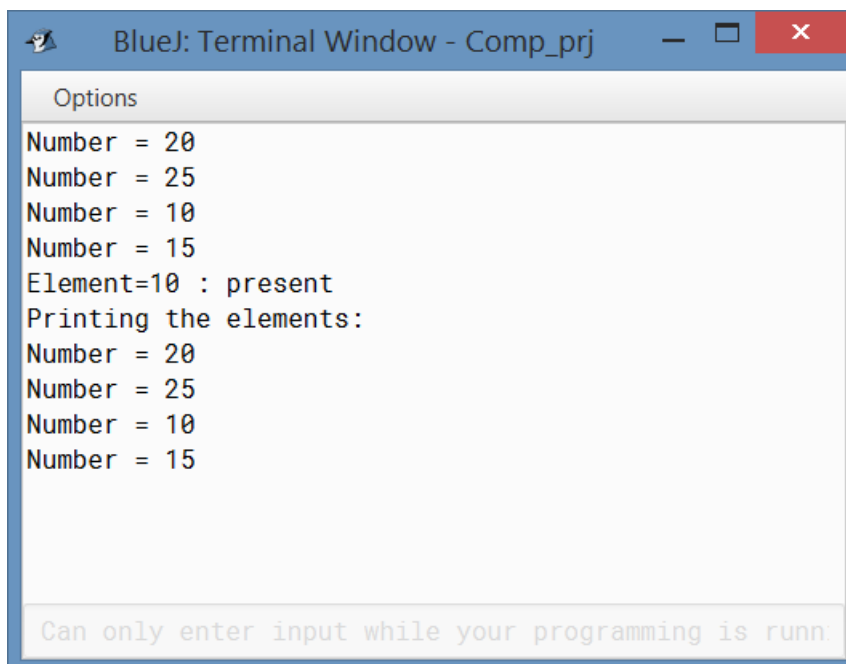
Can only enter input while your programming is running
```

15 ArrayList demonstration

Source code

```
import java.util.*;
class ArrayListDemo
{
    public static void main(String[] args)
    {
        //creates an empty ArrayList
        ArrayList<Integer> arrlist = new ArrayList<Integer>(8);
        arrlist.add(20);
        arrlist.add(25);
        arrlist.add(10);
        arrlist.add(15);
        for(Integer num : arrlist) //extracting elements as Integer
            System.out.println("Number = " + num);
        boolean retval = arrlist.contains(10); //check for number '10'
        if(retval) System.out.println("Element=" + 10 + " : present");
        else System.out.println("Element not found");
        Object[] ob = arrlist.toArray(); //converting to array
        System.out.println("Printing the elements:");
        for(Object val : ob) //extracting elements as Object
            System.out.println("Number = " + val);
    }
}
```

Output



The screenshot shows a terminal window titled "BlueJ: Terminal Window - Comp_prj". The output of the program is as follows:

```
Options
Number = 20
Number = 25
Number = 10
Number = 15
Element=10 : present
Printing the elements:
Number = 20
Number = 25
Number = 10
Number = 15
```

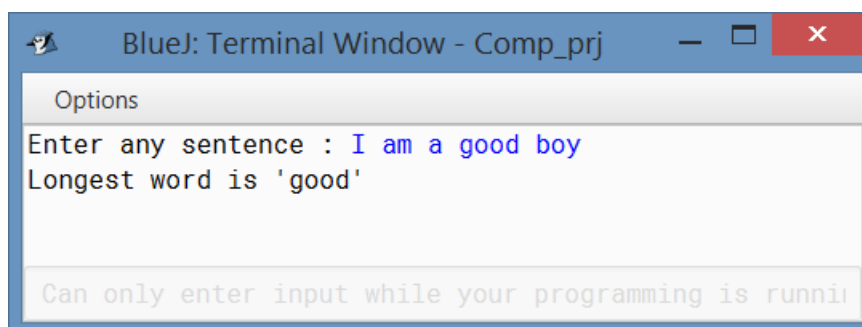
At the bottom of the terminal window, there is a message: "Can only enter input while your programming is running."

16 StringTokenizer demonstration

Source Code

```
// Program to demonstrate StringTokenizer
import java.util.*;
class StringTokenizerDemo
{
    public static void main(String[] args)
    {
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter any sentence : ");
        String str = sc.nextLine();
        StringTokenizer s = new StringTokenizer(str); //StringTokenizer object
        int l = s.countTokens(); //Counts number of tokens with separator = " "
        String longest = "";
        int longestLength = 0;
        for(int i = 0 ; i < l ; i++) //loop to check for longest word
        {
            String t = s.nextToken();
            if(t.length() > longestLength)
            {
                longest = t;
                longestLength = t.length();
            }
        }
        System.out.println("Longest word is '\" + longest+\"'\");
    }
}
```

Output



17 Merge Sort

Source Code

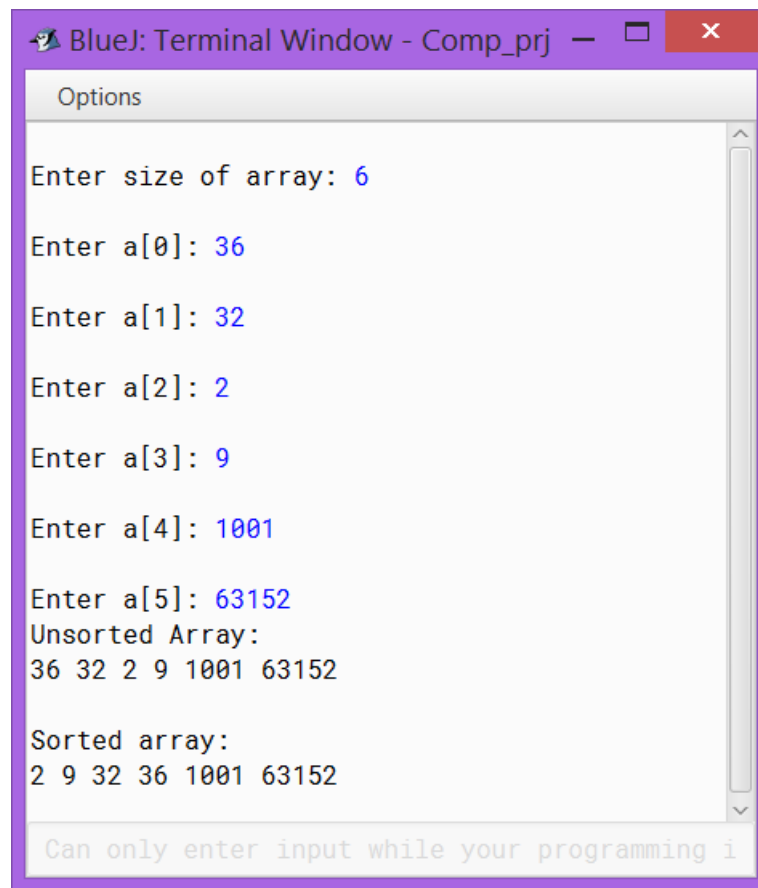
```
// Java program for Merge Sort
import java.io.*;
class MergeSort
{
    // Merges two subarrays of arr[] - conquer
    // First subarray is arr[l..m]
    // Second subarray is arr[m+1..r]
    void merge(int arr[], int l, int m, int r)
    {
        // Find sizes of two subarrays to be merged
        int n1 = m - l + 1;
        int n2 = r - m;
        // Create temporary arrays
        int L[] = new int [n1];
        int R[] = new int [n2];
        //Copy data to temp arrays
        for (int i=0; i<n1; ++i) L[i] = arr[l + i];
        for (int j=0; j<n2; ++j) R[j] = arr[m + 1+ j];
        // Initial indices of first and second subarrays
        int i = 0, j = 0;
        // Initial index of merged subarray array
        int k = l;
        while (i < n1 && j < n2)
        {
            if (L[i] <= R[j])
            {
                arr[k] = L[i];
                i++;
            }
            else
            {
                arr[k] = R[j];
                j++;
            }
            k++;
        }
        // Copy remaining elements of L[]
        while (i < n1)
        {
            arr[k] = L[i];
            i++;
            k++;
        }
        // Copy remaining elements of R[]
        while (j < n2)
        {
            arr[k] = R[j];
            j++;
            k++;
        }
    }
}
```

```

    }
}
// function that sorts arr[l..r] using merge()
void sort(int arr[], int l, int r)
{
    if (l < r)
    {
        int m = (l+r)/2;
        //divide into halves and sort each
        sort(arr, l, m);
        sort(arr , m+1, r);
        // Merge the sorted halves
        merge(arr, l, m, r);
    }
}
//method to print all elements of array
static void printArray(int a[])
{
    int len = a.length;
    for (int i = 0 ; i < len ; i++)
        System.out.print(a[i] + " ");
    System.out.println();
}
// Driver method
public static void main(String args[]) throws IOException
{
    BufferedReader br = new BufferedReader(new InputStreamReader(System.in));
    System.out.print("\nEnter size of array: ");
    int n = Integer.parseInt(br.readLine());
    int[] a = new int[n];
    for(int i = 0 ; i < n ; i++)
    {
        System.out.print("\nEnter a["+i+"]: ");
        a[i] = Integer.parseInt(br.readLine());
    }
    System.out.println("Unsorted Array: ");
    printArray(a);
    MergeSort ob = new MergeSort();
    ob.sort(a, 0, a.length-1);
    System.out.println("\nSorted array: ");
    printArray(a);
}
}

```

Output



The image shows a screenshot of a BlueJ Terminal Window titled "BlueJ: Terminal Window - Comp_prj". The window has a purple title bar and standard window controls. Inside, there is a scrollable text area with the following content:

```
Options

Enter size of array: 6

Enter a[0]: 36

Enter a[1]: 32

Enter a[2]: 2

Enter a[3]: 9

Enter a[4]: 1001

Enter a[5]: 63152
Unsorted Array:
36 32 2 9 1001 63152

Sorted array:
2 9 32 36 1001 63152

Can only enter input while your programming i
```

The input values are shown in blue text. The output shows the unsorted and sorted arrays. The window also features a status bar at the bottom with the message "Can only enter input while your programming i".

18 Quick Sort

Source Code

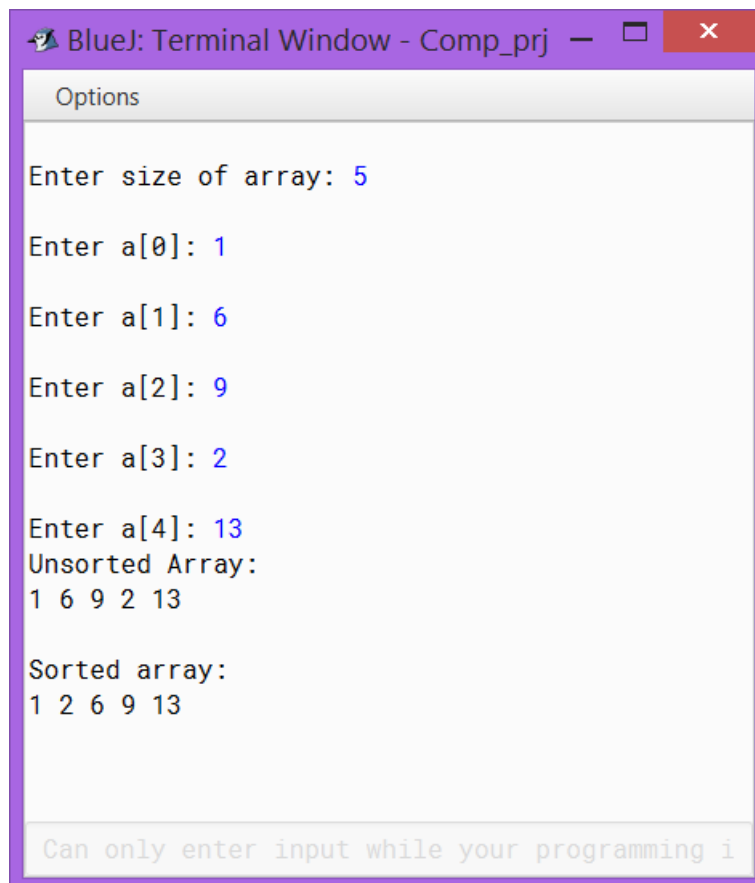
```
// Java program for implementation of QuickSort
import java.io.*;
class QuickSort
{
    //function to partition array
    int partition(int arr[], int low, int high)
    {
        int pivot = arr[high];
        int i = (low-1); // index of smaller element
        for (int j=low; j<high; j++)
        {
            // If current element is smaller than or equal to pivot
            if (arr[j] <= pivot)
            {
                i++;
                // swap arr[i] and arr[j]
                int temp = arr[i];
                arr[i] = arr[j];
                arr[j] = temp;
            }
        }
        // swap arr[i+1] and arr[high] (or pivot)
        int temp = arr[i+1];
        arr[i+1] = arr[high];
        arr[high] = temp;
        return i+1;
    }
    //function to implement quicksort
    void sort(int arr[], int low, int high)
    {
        if (low < high)
        {
            int pi = partition(arr, low, high);
            // Recursively sort elements before partition and after partition
            sort(arr, low, pi-1);
            sort(arr, pi+1, high);
        }
    }
    //method to print all elements of array
    static void printArray(int a[])
    {
        int len = a.length;
        for (int i = 0 ; i < len ; i++)
            System.out.print(a[i] + " ");
        System.out.println();
    }
    // Driver program
    public static void main(String args[]) throws IOException
    {
```

```

BufferedReader br = new BufferedReader(new InputStreamReader(System.in));
System.out.print("\nEnter size of array: ");
int n = Integer.parseInt(br.readLine());
int[] a = new int[n];
for(int i = 0 ; i < n ; i++)
{
    System.out.print("\nEnter a["+i+"]: ");
    a[i] = Integer.parseInt(br.readLine());
}
System.out.println("Unsorted Array: ");
printArray(a);
QuickSort ob = new QuickSort();
ob.sort(a, 0, n-1);
System.out.println("\nSorted array: ");
printArray(a);
}
}

```

Output



```

BlueJ: Terminal Window - Comp_prj
Options
Enter size of array: 5
Enter a[0]: 1
Enter a[1]: 6
Enter a[2]: 9
Enter a[3]: 2
Enter a[4]: 13
Unsorted Array:
1 6 9 2 13

Sorted array:
1 2 6 9 13

Can only enter input while your programming i

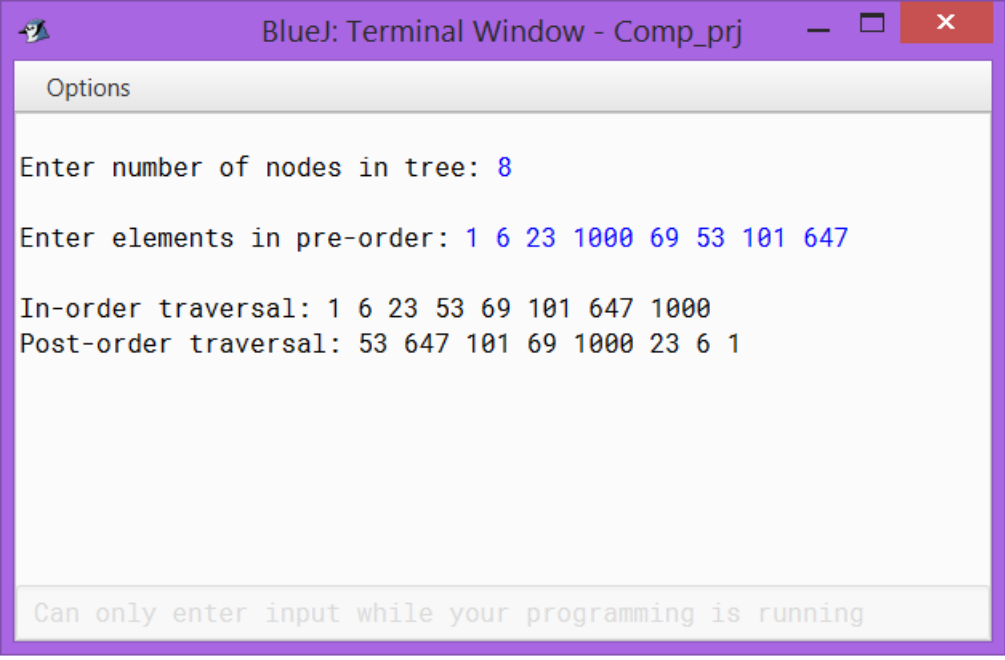
```

19 Traversal in Binary Search Tree

Source Code

```
// Java program to print Postorder traversal from given Preorder traversal
// Binary Search Tree
import java.io.*;
import java.util.*;
public class PrintPost
{
    static int preIndex = 0;
    void printPost(int[] in, int[] pre, int inStart, int inEnd)
    {
        if (inStart > inEnd) return;
        // Find index of next item in preorder traversal in inorder.
        int inIndex = search(in, inStart, inEnd, pre[preIndex++]);
        // traverse left tree
        printPost(in, pre, inStart, inIndex - 1);
        // traverse right tree
        printPost(in, pre, inIndex + 1, inEnd);
        // print root node at the end of traversal
        System.out.print(in[inIndex] + " ");
    }
    int search(int[] in, int startIn, int endIn, int data)
    {
        int i = 0;
        for (i = startIn ; i < endIn ; i++)
            if (in[i] == data) return i;
        return i;
    }
    // Driver method
    public static void main(String args[]) throws IOException
    {
        BufferedReader br = new BufferedReader(new InputStreamReader(System.in));
        System.out.print("\nEnter number of nodes in tree: ");
        int n = Integer.parseInt(br.readLine());
        int[] pre = new int[n];
        int [] in = new int[n];
        System.out.print("\nEnter elements in pre-order: ");
        StringTokenizer st = new StringTokenizer(br.readLine());
        for(int i = 0 ; i < n ; i++)
        {
            pre[i] = Integer.parseInt(st.nextToken());
            in[i] = pre[i];
        }
        Arrays.sort(in);
        System.out.print("\nIn-order traversal: ");
        for(int i = 0 ; i < n ; i++) System.out.print(in[i]+" ");
        PrintPost tree = new PrintPost();
        System.out.print("\nPost-order traversal: ");
        tree.printPost(in, pre, 0, n - 1);
    }
}
```

Output



A screenshot of a BlueJ terminal window titled "BlueJ: Terminal Window - Comp_prj". The window has a purple title bar with standard window controls. Below the title bar is a tab labeled "Options". The main area of the terminal displays the following text:

```
Enter number of nodes in tree: 8  
Enter elements in pre-order: 1 6 23 1000 69 53 101 647  
In-order traversal: 1 6 23 53 69 101 647 1000  
Post-order traversal: 53 647 101 69 1000 23 6 1
```

At the bottom of the terminal window, there is a light gray bar with the text "Can only enter input while your programming is running".