1

```
import java.util.Scanner;
public class Student {
   private String usn, name, branch, phone;
   public Student(String usn, String name, String branch,
String phone) {
      super();
      this.usn = usn;
      this.name = name;
      this.branch = branch;
      this.phone = phone;
   }
   public String toString() {
      return "Student [USN = " + usn + ", NAME = " +
name + ", BRANCH = " + branch
            + ", PHONE NUMBER = " + phone + "]";
   }
   public static void main(String args[]) {
      int i;
```

```
        String usn, branch, name, phone;
        Scanner s = new Scanner(System.in);
        System.out.println("Enter number of Students: ");
        int n = s.nextInt();
        Student[] students = new Student[n + 1];
        for (i = 1; i <= n; i++) {
            System.out.println("Enter student " + i + "
details\n");
            System.out.println("Give Student Details USN,
Name, Branch, Phone Number");
            usn = s.next();
            name = s.next();
            branch = s.next();
            phone = s.next();
            s.close();
            students[i] = new Student(usn, name, branch,
phone);
        }
        System.out.println("DATABASE:");
```

```
        for (i = 1; i <= n; i++) {
            System.out.println(students[i]);
        }
    }
}
```

OUTPUT:

Enter number of Students:

2

Enter student 1 details

Give Student Details USN, Name, Branch, Phone Number

4BD13CV001

ARJUN

CIVIL

9264921640

Enter student 2 details

Give Student Details USN, Name, Branch, Phone Number

4BD15IS010

CHARAN

IS

7592783640

DATABASE:

Student [USN = 4BD13CV001, NAME = ARJUN, BRANCH = CIVIL, PHONE NUMBER = 9264921640]

Student [USN = 4BD15IS010, NAME = CHARAN, BRANCH = IS, PHONE NUMBER = 7592783640]

2

Stack

```
import java.util.Scanner;
public class StackDemo {
    public static void main(String[] args) {
        int top = -1;
        int n, element, i;
        int[] a;
        Scanner s = new Scanner(System.in);
        System.out.println("Enter Stack Size");
        n = s.nextInt();
        a = new int[n + 1];
        System.out.println("Stack operations are" + "\t" +
"1.Push" + "\t" + "2.POP" + "\t" +
                "3.Display" + "\t" + "4.Exit");
        for (;;) {
            System.out.println("Enter your choice");
            int choice = s.nextInt();
            switch (choice) {
                case 1:
                    if (top == n - 1) {
```

```
                System.out.println("Stack overflow");
            } else {
                System.out.println("Enter element to be
pushed");
                element = s.nextInt();
                a[top++] = element;
            }
            break;
        case 2:
            if (top == -1) {
                System.out.println("Stack Underflow");
            } else {
                System.out.println("Popped element " +
a[top--]);
            }
            break;
        case 3:
            if (top == -1) {
                System.out.println("Stack Empty");
```

```
                } else {
                    System.out.println("Elements in stack :");
                    for (i = top; i >= 0; i--) {
                        System.out.println(a[i]);
                    }
                }
                break;
            case 4:
                System.exit(0);
                break;
        }
    }
}
```

OUTPUT:

```
Enter Stack Size
3
Stack operations are 1.Push 2.POP 3.Display 4.Exit
Enter your choice
```

1

Enter element to be pushed

53

Enter your choice

1

Enter element to be pushed

68

Enter your choice

1

Enter element to be pushed

20

Enter your choice

1

Stack overflow

Enter your choice

3

Elements in stack :

20

68

53

Enter your choice

2

Popped element 20

Enter your choice

2

Popped element 68

Enter your choice

2

Popped element 53

Enter your choice

2

Stack Underflow

Enter your choice

3

Stack Empty

Enter your choice

4

3

Customer

```
import java.util.Scanner;
import java.util.StringTokenizer;
class Customer {
    private String customerName, date;
```

```
    public Customer(String customerName, String date) {
        super();
        this.customerName = customerName;
        this.date = date;
    }
    public String toString() {
        String returnValue = customerName + ",";
        StringTokenizer tokenizer = new
StringTokenizer(date, "/");
        System.out.println("The Customer details are ");
        while (tokenizer.hasMoreTokens()) {
            returnValue = returnValue +
tokenizer.nextToken();
            if (tokenizer.hasMoreElements()) {
                returnValue = returnValue + ",";
            }
        }
        return returnValue;
    }
```

```
}
public class CustomerDetails {
   public static void main(String[] args) {
      String customerName;
      String date;
      Scanner scanner = new Scanner(System.in);
      System.out.println("Enter customer name");
      customerName = scanner.next();
      System.out.println("Enter Date (dd/mm/yyy)");
      date = scanner.next();
      scanner.close();
      Customer customer = new
Customer(customerName, date);
      System.out.println(customer.toString());
   }
}
```

OUTPUT:

```
Enter customer name
Thomas
```

Enter Date (dd/mm/yyy)

10/10/1916

The Customer details are

Thomas,10,10,1916

4

```
public class exp4 {
   public static void main(String[] args) {
      int n, a[], i;
      Scanner input = new Scanner(System.in);
      System.out.println("Enter the Size of an Array: ");
      n = input.nextInt();
      input.close();
      a = new int[n + 1];
      Random rn = new Random();
      System.out.println("System automatically
generates numbers ");
      for (i = 0; i < n; ++i) {
```

```
    a[i] = rn.nextInt(n);// a[i] = input.nextInt();
}
a[i] = 100000; // Sentinel value
MergeSort mSort = new MergeSort(a);
System.out.println("Before Sort: ");
for (i = 0; i < n; ++i) {
    System.out.print(a[i] + "\t");
}
int low = 0;
int high = n - 1;
mSort.mergeSort(low, high);
System.out.println("\n\nAfter Sort: ");
for (i = 0; i < n; ++i) {
    System.out.print(a[i] + "\t");
}
int step = 2000;
double duration;
/* times for n = 0, 10, ..., 100, 200, ..., 5000 */
System.out.println("\n\nN\tRepetitions\tTime\n");
```

```
        for (n = 5000; n < 50000; n += step) {
            a = new int[n + 1];
            mSort = new MergeSort(a);
            /* get time for size n */
            long repetitions = 0;
            long start = System.nanoTime();
            do {
                repetitions++;
                for (i = 0; i < n; ++i) {
                    a[i] = rn.nextInt(n);
                }

                a[i] = 100000; // Sentinel value
                mSort.mergeSort(0, n - 1);
            } while (System.nanoTime() - start <
1000000000);


            /* repeat until enough time has elapsed */
```

```
        duration = ((double) (System.nanoTime() - start))
/ 1000000000;
        duration /= repetitions;
        System.out.println(n + "\t" + repetitions + "\t\t"
+ duration);
      }
   }
}
```

Enter the Size of an Array:

4

System automatically generates numbers

Before Sort:

3 3 3 3

After Sort:

3 3 3 3

N	Repetitions	Time
5000	71	0.0141507802816901 4
7000	51	0.0197775862745098 07
9000	48	0.0210191208333333 35

11000	35	0.02880425714285715
13000	25	0.04043286
15000	17	0.0589115941176470 56
17000	15	0.0688320733333334
19000	12	0.0888746916666667
21000	9	0.1111643666666668
23000	8	0.1374092125
25000	8	0.1357298625
27000	7	0.1558221714285714 4
29000	6	0.1747082666666666 4
31000	5	0.20674228
33000	4	0.2687321
35000	4	0.2606524
37000	4	0.290150675
39000	4	0.3047878
41000	3	0.3933640333333333
43000	3	0.3999323666666667
45000	3	0.4472211333333335
47000	3	0.497967

49000 3 0.46362176666666666

5

A) Insertion sort

```
class InsertionSort {
   void sort(int arr[]) {
      int n = arr.length;
      for (int i = 1; i < n; ++i) {
         int key = arr[i];
         int j = i - 1;
         while (j >= 0 && arr[j] > key) {
            arr[j + 1] = arr[j];
            j = j - 1;
         }
         arr[j + 1] = key;
      }
   }
   void printArray(int arr[]) {
      int n = arr.length;
```

```
        for (int i = 0; i < n; ++i) {
            System.out.print(arr[i] + " ");
        }
        System.out.println();
    }
}
public class exp5_a {
    public static void main(String args[]) {
        int arr[] = { 12, 11, 13, 5, 6 };
        InsertionSort ob = new InsertionSort();
        ob.sort(arr);
       ob.printArray(arr);
    }
}
```

Output : 5 6 11 12 13

B) Selection sort

```
class SelectionSort {
    void sort(int arr[]) {
```

```
    int n = arr.length;
    for (int i = 0; i < n - 1; i++) {
        int min_idx = i;
        for (int j = i + 1; j < n; j++) {
            if (arr[j] < arr[min_idx]) {
                min_idx = j;
            }
        }
        int temp = arr[min_idx];
        arr[min_idx] = arr[i];
        arr[i] = temp;
    }
}
void printArray(int arr[]) {
    int n = arr.length;
    for (int i = 0; i < n; ++i) {
        System.out.print(arr[i] + " ");
    }
    System.out.println();
```

```
    }
}
public class exp5_b {
    public static void main(String args[]) {
        SelectionSort ob = new SelectionSort();
        int arr[] = { 64, 25, 12, 22, 11 };
        ob.sort(arr);
        System.out.println("Sorted array");
        ob.printArray(arr);
    }
}
```

Output:

Sorted array

11 12 22 25 64

6)

```
import java.util.*;
public class exp6 {
    public static void main(String args[]) {
```

```
Scanner sc = new Scanner(System.in);
System.out.println("Enter the number of Jobs");
int n = sc.nextInt();
String a[] = new String[n];
int b[] = new int[n];
int c[] = new int[n];
for (int i = 0; i < n; i++) {
   System.out.println("Enter the Job"+(i+1));
   a[i] = sc.next();
   System.out.println("Enter the Profit");
   b[i] = sc.nextInt();
   System.out.println("Enter the DeadLine");
   c[i] = sc.nextInt();
}
sc.close();
System.out.println("--Arranged Order--");
System.out.print("Jobs:    ");
for (int i = 0; i < n; i++) {
   System.out.print(a[i] + " ");
```

```
}
System.out.println();
System.out.print("Profit:  ");
for (int i = 0; i < n; i++) {
   System.out.print(b[i] + " ");
}
System.out.println();
System.out.print("DeadLine:");
for (int i = 0; i < n; i++) {
   System.out.print(c[i] + " ");
}
for (int i = 0; i < n - 1; i++) {
   for (int j = i + 1; j < n; j++) {
      if (b[i] < b[j]) {
         int temp = b[i];
         b[i] = b[j];
         b[j] = temp;
         temp = c[i];
         c[i] = c[j];
```

```
            c[j] = temp;
            String temp1 = a[i];
            a[i] = a[j];
            a[j] = temp1;
        }
    }
}
System.out.println();
System.out.println("--Sorted Order--");
System.out.print("Jobs:    ");
for (int i = 0; i < n; i++) {
    System.out.print(a[i] + " ");
}
System.out.println();
System.out.print("Profit:  ");
for (int i = 0; i < n; i++) {
    System.out.print(b[i] + " ");
}
System.out.println();
```

```
System.out.print("DeadLine:");
for (int i = 0; i < n; i++) {
   System.out.print(c[i] + " ");
}
System.out.println();
int max = c[0];
for (int i = 0; i < n; i++) {
   if (c[i] > max) {
      max = c[i];
   }
}
String x[] = new String[max];
// int xx[] = new int[max];
int profit = 0;
for (int i = 0; i < n; i++) {
   int pp = c[i];
   pp = pp - 1;
   if (x[pp] == null) {
      x[pp] = a[i];
```

```
            profit += b[i];
        } else {
            while (pp != -1) {
                if (x[pp] == null) {
                    x[pp] = a[i];
                    profit += b[i];
                    break;
                }
                pp = pp - 1;
            }
        }
    }
    for (int i = 0; i < max; i++) {
        System.out.print("-->" + x[i]);
    }
    System.out.println();
    System.out.print("Profit Earned:" + profit);
  }
}
```

Output:

Enter the number of Jobs

2

Enter the Job1

1

Enter the Profit

2

Enter the DeadLine

3

Enter the Job2

2

Enter the Profit

3

Enter the DeadLine

4

--Arranged Order--

Jobs: 1 2

Profit: 2 3

DeadLine:3 4

--Sorted Order--

Jobs: 2 1

Profit: 3 2

DeadLine:4 3

-->null-->null-->1-->2

Profit Earned:5

9

```
class exp9 {
static int N = 8;
static boolean isSafe(int x, int y, int sol[][])
{
return (x >= 0 && x < N && y >= 0 && y < N
&& sol[x][y] == -1);
}
static void printSolution(int sol[][])
{
for (int x = 0; x < N; x++) {
for (int y = 0; y < N; y++)
```

```
System.out.print(sol[x][y] + " ");
System.out.println();
}
}
static boolean solveKT()
{
int sol[][] = new int[8][8];
for (int x = 0; x < N; x++)
for (int y = 0; y < N; y++)
sol[x][y] = -1;
int xMove[] = { 2, 1, -1, -2, -2, -1, 1, 2 };
int yMove[] = { 1, 2, 2, 1, -1, -2, -2, -1 };
sol[0][0] = 0;
if (!solveKTUtil(0, 0, 1, sol, xMove, yMove)) {
System.out.println("Solution does not exist");
return false;
}
else
printSolution(sol);
```

```
return true;
}
static boolean solveKTUtil(int x, int y, int movei,
int sol[][], int xMove[],
int yMove[])
{
int k, next_x, next_y;
if (movei == N * N)
return true;
for (k = 0; k < 8; k++) {
next_x = x + xMove[k];
next_y = y + yMove[k];
if (isSafe(next_x, next_y, sol)) {
sol[next_x][next_y] = movei;
if (solveKTUtil(next_x, next_y, movei + 1,
sol, xMove, yMove))
return true;
```

```
else
sol[next_x][next_y]
= -1;
}
}
return false;
}
public static void main(String args[])
{
// Function Call
solveKT();
}
}
```

Output:

```
0 59 38 33 30 17 8 63
37 34 31 60 9 62 29 16
58 1 36 39 32 27 18 7
35 48 41 26 61 10 15 28
42 57 2 49 40 23 6 19
```

47 50 45 54 25 20 11 14

56 43 52 3 22 13 24 5

51 46 55 44 53 4 21 12

10

A)

```
import java.util.ArrayList;
class SubSet_sum_problem {
   static boolean[][] dp;
   static void display(ArrayList<Integer> v) {
      System.out.println(v);
   }
   static void printSubsetsRec(int arr[], int i, int sum,
         ArrayList<Integer> p) {
      if (i == 0 && sum != 0 && dp[0][sum]) {
         p.add(arr[i]);
         display(p);
         p.clear();
```

```
        return;
    }
    if (i == 0 && sum == 0) {
        display(p);
        p.clear();
        return;
    }
    if (dp[i - 1][sum]) {
        ArrayList<Integer> b = new ArrayList<>();
        b.addAll(p);
        printSubsetsRec(arr, i - 1, sum, b);
    }
    if (sum >= arr[i] && dp[i - 1][sum - arr[i]]) {
        p.add(arr[i]);
        printSubsetsRec(arr, i - 1, sum - arr[i], p);
    }
}
void printAllSubsets(int arr[], int n, int sum) {
    if (n == 0 || sum < 0)
```

```
    return;
dp = new boolean[n][sum + 1];
for (int i = 0; i < n; ++i) {
    dp[i][0] = true;
}
if (arr[0] <= sum)
    dp[0][arr[0]] = true;
for (int i = 1; i < n; ++i)
    for (int j = 0; j < sum + 1; ++j)
        dp[i][j] = (arr[i] <= j) ? (dp[i - 1][j] ||
                dp[i - 1][j - arr[i]])
                : dp[i - 1][j];
if (dp[n - 1][sum] == false) {
    System.out.println("There are no subsets with" +
            " sum " + sum);
    return;
}
ArrayList<Integer> p = new ArrayList<>();
printSubsetsRec(arr, n - 1, sum, p);
```

```
    }
}
public class exp10_a {
    public static void main(String args[]) {
        int arr[] = { 1, 2, 3, 4, 5 };
        int n = arr.length;
        int sum = 10;
        SubSet_sum_problem ob = new
SubSet_sum_problem();
        ob.printAllSubsets(arr, n, sum);
    }
}
```

Output:

```
[4, 3, 2, 1]
[5, 3, 2]
[5, 4, 1]
```

B)

```
class Permutation
{
public void permute(String str, int l, int r)
{
if (l == r)
System.out.println(str);
else
{
for (int i = l; i <= r; i++)
{
str = swap(str,l,i);
permute(str, l+1, r);
str = swap(str,l,i);
}
}
}
public String swap(String a, int i, int j)
{
char temp;
```

```
char[] charArray = a.toCharArray();
temp = charArray[i] ;
charArray[i] = charArray[j];
charArray[j] = temp;
return String.valueOf(charArray);
}
}
public class exp10_b {
public static void main(String[] args) {
String str = "ABC";
int n = str.length();
Permutation permutation = new Permutation();
permutation.permute(str, 0, n - 1);
}
}
```

Output:

```
ABC
```

ACB

BAC

BCA

CBA

CAB