- **Arithmetic Calculator**

```
public class Arithmeticcalculator {
 public static void main(String[] args) {
            Scanner sc=new Scanner(System.in);
            System.out.println("Enter the two numbers");
            int firstnum=sc.nextInt();
            int secnum= sc.nextInt();
            System.out.println("Enter the operator ");
            char op=sc.next().charAt(0);
            double output=0;

            switch(op){
            case '+': output=firstnum+secnum;
            break;
            case '-':output=firstnum-secnum;
            break;
            case '*': output=firstnum*secnum;
            break;
            case '/':output=firstnum/secnum;
            break;
            }
            System.out.println("the answer is " + output);
      }
}
```

- **Validation of Email ID**

```
          public class validationemail {
             public static boolean isValidEmail(String email) {
                            String regex = "^(.+)@(.+)$";

                           Pattern pattern = Pattern.compile(regex);
                           Matcher matcher = pattern.matcher(email);
                           return matcher.matches();
                         }

                        public static void main(String[] args) {
                            List<String> emails = new ArrayList<String>();
                            // valid email addresses
                            emails.add("arun@gmail.com");
                            emails.add("ganesh@gmail.com");
                            emails.add("ramu@egmail.me.org");
                            emails.add("ravi@gmail.me.org");
                            //invalid email addresses
                            emails.add("amul.example.com");
                            emails.add("akhil..reddy.com");
                            emails.add("akshay.project.com");

                       System.out.println("Enter any email address to check");
                            Scanner sc = new Scanner(System.in);
                            String input = sc.nextLine();
           System.out.println("The Email address " + input + " is " +
          (isValidEmail(input) ? "valid" : "invalid"));

                        }

          }
```

- **File Handling**

```
public class Filehandling {

	public static void main(String[] args) {
		Scanner sc = new Scanner(System.in);
		 String choice,cont = "y";

		 while( cont.equalsIgnoreCase("y") ) {
			  System.out.println("\t\t cricket player \n\n");

			System.out.println("a ->Add New player Record ");
			System.out.println("b -> View All player Record ");
			System.out.println("c -> Delete player Record ");
			System.out.println("d -> Search player Record ");
			System.out.println("e -> Update player Record ");

			System.out.print("\n\n");
			System.out.println("Enter your choice: ");
			choice = sc.nextLine();

			if( choice.equals("a") ) {
				try {
					AddRecord();
				} catch (IOException e) {

					e.printStackTrace();
				}
			} else if( choice.equals("b") ) {
				try {
					ViewAllRecord();
				} catch (IOException e) {

					e.printStackTrace();
				}
			} else if( choice.equals("c") ) {
				try {
					DeleteRecordByID();
				} catch (IOException e) {

					e.printStackTrace();
				}
			}    else if( choice.equals("d") ) {
				try {
					SearchRecordbyID();
				} catch (IOException e) {

					e.printStackTrace();
				}
			}    else if( choice.equals("e") ) {
				try {
					updateRecordbyID();
				} catch (IOException e) {

					e.printStackTrace();
				}
			}

			System.out.println("Do you want to continue? Y/N");
```

```
                        cont = sc.nextLine();

                    }

            }

              public static void AddRecord() throws IOException {

BufferedWriter bw = new BufferedWriter( new FileWriter("records.txt",true) );

                    Scanner sc1 = new Scanner(System.in);

                    String ID, name, age, addr;

                    System.out.print("Enter the player ID: ");
                    ID = sc1.nextLine();
                    System.out.print("Enter the player Name: ");
                    name = sc1.nextLine();
                    System.out.print("Enter the player Age: ");
                    age = sc1.nextLine();
                    System.out.print("Enter the player Address: ");
                    addr = sc1.nextLine();

                    bw.write(ID+","+name+","+age+","+addr);
                    bw.flush();
                    bw.newLine();
                    bw.close();

        }


              public static void ViewAllRecord() throws IOException {
BufferedReader br = new BufferedReader( new FileReader("records.txt") );

                    String record;

      System.out.println("|     ID           Name   Age          Address |");

                    while( ( record = br.readLine() ) != null ) {

                          StringTokenizer st = new StringTokenizer(record,",");

System.out.println("|  "+st.nextToken()+"  "+st.nextToken()+"
   "+st.nextToken()+"              "+st.nextToken()+"      |");

                    }

                    br.close();

                }


              public static void DeleteRecordByID() throws IOException {
                          Scanner sc2 =  new Scanner(System.in);
                          String ID, record;


                          File tempDB = new File("records_temp.txt");
                          File db = new File("records.txt");
```

```
  BufferedReader br = new BufferedReader( new FileReader( db ) );
  BufferedWriter bw = new BufferedWriter( new FileWriter( tempDB ) );


                        System.out.println("\t\t Delete player Record\n");

                        System.out.println("Enter the player ID: ");
                        ID =  sc2.nextLine();


                        while( ( record = br.readLine() ) != null ) {


                              if( record.contains(ID) )
                                    continue;

                              bw.write(record);
                              bw.flush();
                              bw.newLine();

                        }

                        br.close();
                        bw.close();

                        db.delete();

                        tempDB.renameTo(db);

                }


          public static void SearchRecordbyID() throws IOException {
                        String ID,record;
                        Scanner sc3 = new Scanner(System.in);

    BufferedReader br = new BufferedReader( new FileReader("records.txt") );

                        System.out.println("\t\t Search player Record\n");


                        System.out.println("Enter the player ID: ");
                        ID = sc3.nextLine();

System.out.println("|  ID           Name   Age          Address  |");

                        while( ( record = br.readLine() ) != null ) {

                StringTokenizer st = new StringTokenizer(record,",");

          if( record.contains(ID) ) {
System.out.println("|  "+st.nextToken()+"  "+st.nextToken()+"
   "+st.nextToken()+"               "+st.nextToken()+"      |");
                              }

                        }
                        br.close();
                  }
```

```
        public static void updateRecordbyID() throws IOException {
                        String newName, newAge, newAddr, record, ID,record2;

                        File db = new File("records.txt");
                        File tempDB = new File("records_temp.txt");

        BufferedReader br = new BufferedReader( new FileReader(db) );
        BufferedWriter bw = new BufferedWriter( new FileWriter(tempDB) );

                        Scanner sc4 = new Scanner(System.in);

                        System.out.println("\t\t Update player Record\n\n");

                                System.out.println("Enter the player ID: ");
                                ID = sc4.nextLine();
    System.out.println("|      ID           Name   Age          Address |");

                                while( ( record = br.readLine() ) != null ) {

                    StringTokenizer st = new StringTokenizer(record,",");
                       if( record.contains(ID) ) {
                     System.out.println("|"+st.nextToken()+"     "+st.nextToken()+"
    "+st.nextToken()+"  "+st.nextToken()+"|");
                                        }

                                }

                        br.close();

                        System.out.println("Enter the new Name: ");
                        newName = sc4.nextLine();
                        System.out.println("Enter the new Age: ");
                        newAge = sc4.nextLine();
                        System.out.println("Enter the new Address: ");
                        newAddr = sc4.nextLine();

    BufferedReader br2 = new BufferedReader( new FileReader(db) );

                        while( (record2 = br2.readLine() ) != null ) {

                                if(record2.contains(ID)) {

  bw.write(ID+","+newName+","+newAge+","+newAddr);
                                } else {

                                bw.write(record2);
                                }
                                bw.flush();
                                bw.newLine();
                        }

                        bw.close();
                        br2.close();
                        db.delete();
                        boolean success = tempDB.renameTo(db);
                        System.out.println(success);

                }

    }
```

- **Longest increasing Subsequence**

```
public class Longestincreasingsubsequence {

	static int max_ref;

	static int _lis(int arr[], int n)
	{

		if (n == 1)
			return 1;

		int res, max_ending_here = 1;


		for (int i = 1; i < n; i++)
		{
			res = _lis(arr, i);
			if (arr[i - 1] < arr[n - 1]
				&& res + 1 > max_ending_here)
				max_ending_here = res + 1;
		}


		if (max_ref < max_ending_here)
			max_ref = max_ending_here;


		return max_ending_here;
	}

	static int lis(int arr[], int n)
	{

		max_ref = 1;
		_lis(arr, n);
		return max_ref;
	}

	public static void main(String args[])
	{
		int arr[] = { 10,20, 12, 8, 23, 22, 40, 51 };
		int n = arr.length;
		System.out.println("Length of lis is " + lis(arr, n)+ "\n");
	}
}
```

- **Fix Bugs of Application**

```
public class Bugfix {
public static void main(String[] args) { System.out.println("\t Welcome to TheDesk
\n"); optionsSelection();
}
private static void optionsSelection() {
```

```
String[] arr = {"1. I wish to review my expenditure",
                      "2. I wish to add my expenditure",
                "3. I wish to delete my expenditure",
                "4. I wish to sort the expenditures",
                "5. I wish to search for a particular expenditure",
                "6. Close the application"
};
int[] arr1 = {1,2,3,4,5,6};
int slen = arr1.length;
for(int i=0; i<slen;i++){ System.out.println(arr[i]);
}
ArrayList<Integer> arrlist = new ArrayList<Integer>();
ArrayList<Integer> expenses = new ArrayList<Integer>();
expenses.add(10000);
expenses.add(200);
expenses.add(5000);
expenses.add(12000);
expenses.add(1100);
expenses.addAll(arrlist);
System.out.println("\nEnter your choice:\t"); Scanner sc = new Scanner(System.in);
int options = sc.nextInt();
for(int j=1;j<=slen;j++){
if(options==j){
switch (options){
case 1:
System.out.println("Your saved expenses are listed below: \n");
System.out.println(expenses+"\n");
optionsSelection();
break;
case 2:
System.out.println("Enter the value to add your Expense: \n");
int value = sc.nextInt(); expenses.add(value);
System.out.println("Your value is updated\n");
expenses.addAll(arrlist);
System.out.println(expenses+"\n");
optionsSoptionsSelection();
break;
case 3:
System.out.println("You are about the delete all your expenses! \nConfirm again by
selecting the same option...\n");
int con_choice=sc.nextInt();
if(con_choice==options)
{
expenses.clear();
System.out.println(expenses+"\n");
System.out.println("All your expenses are erased!\n");
} else {
System.out.println("Oops... try again!");
}
optionsSelection();
break;
case 4:
sortExpenses(expenses); optionsSelection();
break;
case 5:
searchExpenses(expenses); optionsSelection();
break;
case 6:
closeApp();
break;
```

```
default:
System.out.println("You have made an invalid choice!");
break;
}
}
}

}
private static void closeApp() {
System.out.println("Closing your application... \nThank you!");
}
private static void searchExpenses(ArrayList<Integer> arrayList) {
int leng = arrayList.size();
System.out.println("Enter the expense you need to search:\t");

Scanner sc = new Scanner(System.in);
int input = sc.nextInt();
//Linear Search
for(int i=0;i<leng;i++) {
if(arrayList.get(i)==input) {
System.out.println("Found the expense " + input + " at " +i + " position");
}
}
}
private static void sortExpenses(ArrayList<Integer> arrayList) {
int arrlength = arrayList.size();
//Complete the method. The expenses should be sorted in ascending order.

Collections.sort(arrayList);
System.out.println("Sorted expenses: ");
for(Integer i: arrayList) {
System.out.print(i + " ");
}

System.out.println("\n");

}
}
```