METHODS (FUNCTIONS)

Chapter - 9

Type A: Very Short/ Short Answer Questions

1	**Method not returning any value has return type as:** **(a) int (b) char** **(c) float (d) void.**
Ans.	(d) void
2	**A method can return_____values.** **(a) 1 (b) 2** **(c) 3 (d) all of these.**
Ans.	1
3	**The parameters appearing in method call statement are called** **(a) actual parameter (b) formal parameter** **(c) call parameter (d) all of the above**
Ans.	(a) actual parameter
4	**The parameters appearing in method definition are called** **(a) actual parameter (b) formal parameter** **(c) call parameter (d) all of the above**
Ans.	(b) formal parameter
5	**The method call in which the data in actual parameters remains intact is known as** **(a) call by value (b) call by reference** **(c) return by value (d) return by reference.**
Ans.	(a) call by value
6	**The method call in which the data in actual parameters gets changed is known as** **(a) call by value (b) call by reference** **(c) Return by value (d) Return by reference.**
Ans.	(b) call by reference
7	**The method that changes the state of its parameters is called** **(a) pure method (b) impure method** **(c) change method (d) none of above.**
Ans.	None of the above
8	**One method, many definitions, is called** **(a) method enlargement (b) method overloading** **(c) method loading (d) all of above.**
Ans.	(b) method overloading
9	**Which of the following is not an advantage of methods?** **(a) it helps cope up complexity in programs** **(b) it makes subprogram reusable** **(c) it hides the implementation details** **(d) it offers mathematical solution of problems.**
Ans.	(d) it offers mathematical solution of problems.
10	**Fill in the blanks** **(i) In java, methods reside in _____.** **(ii) The number and type of arguments of a method are known as _____.** **(iii) The first line of method definition that tells about the type of return value along with number and type of arguments is called _____.** **(iv) A member method having the same name as that of its class is called _____ method.** **(v) A constructor method has _____ return type.** **(vi) A private constructor allows object creation only inside _____methods.** **(vii) A _____ constructor takes no arguments.**

	(viii)	**A _____constructor creates objects through values passed to it.**
	(ix)	**The keyword _____ refers to current object.**
Ans.	(i)	In java, methods reside in **classes.**
	(ii)	The number and type of arguments of a method are known as **parameter list.**
	(iii)	The first line of method definition that tells about the type of return value along with number and type of arguments is called **method prototype.**
	(iv)	A member method having the same name as that of its class is called **constructor** method.
	(v)	A constructor method has **no** return type.
	(vi)	A private constructor allows object creation only inside **private** methods.
	(vii)	A **non-parameterized** constructor takes no arguments.
	(viii)	A **parameterized** constructor creates objects through values passed to it.
	(ix)	The keyword **this** refers to current object.
11	**State True or False:** **(a) A method argument is a value returned by the method to the calling program.** **(b) A method declared as static cannot access non-static class members.** **(c) A static class method can be invoked by simply using the name of the method alone.**	
Ans.	False True True	
12	**What is the principal reason for passing arguments by value?**	
Ans.	The principal reason for passing arguments by value is that you cannot alter the variables that are used to call the method because any change that occurs inside called method is on the method's copy of the argument value.	
13	**When an argument is passed by reference,** **(a) A variable is created in the method to hold the argument's value.** **(b) The method cannot access the argument's value.** **(c) A temporary variable is created in the calling program to hold the argument's value.** **(d) The method accesses the argument's original value in the calling program.**	
Ans.	**(d)** The method accesses the argument's original value in the calling program.	
14	**What is the role of return statement in a method?**	
Ans.	The return statement is used to return a value to the calling method. When a return statement does not return a value it returns the control to the calling method.	
15	**What are three types of methods in Java?**	
Ans.	Computational Methods Procedural Methods: Manipulative Methods	
16	**At what time is the constructor method automatically invoked?**	
Ans.	The constructor method is automatically invoked when an object is created class.	

Type B: Short Answer Questions

1	**Define a method. What is method prototype?**
Ans.	**Methods** are functions that operate on instances of classes in which they are defined. Objects can communicate with each other using methods and can call methods in other classes. Method definition has four parts. They are name of the method; type of object or primitive type the method returns, a list of parameters and the body of the method. **Method Prototype** tells the compiler about the type of the value returned by the method, the number and type of arguments. For example, int absval(int a);
2	**What are actual and formal parameters of a method?**
Ans.	**Actual parameter**: A variable or expression contained in a method call and passed to that method is called actual parameter. **Formal parameter:** A name, introduced in a method definition that is replaced by an actual parameter when the

<table>
<tr><td></td><td>method is called.</td></tr>
<tr><td>3</td><td>How many values can be returned from a method?</td></tr>
<tr><td>Ans.</td><td>One value can be returned from a method.</td></tr>
<tr><td>4</td><td>Identify the errors in the method skeletors given below:
(i) float average (a, b) { }
(ii) float mult (int x, y) { }
(iii) float doer (int, float = 3.14) { }</td></tr>
<tr><td>Ans.</td><td>i. Data types are missing in formal parameters
i. Parameter y doesn't have data type
i. int data type doesn't have parameter name</td></tr>
<tr><td>5</td><td>Given the method below write a main() method that includes everything necessary to call this method.
int thrice (int x)
{ return a*3; }</td></tr>
<tr><td>Ans.</td><td><pre>public static void main(String args[])
{
      int result=thrice(3);
}</pre></td></tr>
<tr><td>6</td><td>What is the principal reason for passing arguments by value? What is the principal reason for passing arguments by reference? In a method call, what all data items can be passed by reference?</td></tr>
<tr><td>Ans.</td><td>The main benefit of passing arguments by value is that alterations in variables are not possible. This method is very useful in situation when to ensure that the value being passed remains unaltered in the caller method.
The principal reason for passing arguments by reference is to work on single copy of values of variables by this mechanism changes in one variable reflected on its respective variable. Passing arguments by reference are useful when the values of the original variables are to be changed using the method.
All reference type data like array, object can be passed by reference.</td></tr>
<tr><td>7</td><td>Differentiate between CALL by reference and CALL by value.</td></tr>
<tr><td>Ans.</td><td>
<table>
<tr><th>Call by reference</th><th>Call By Value</th></tr>
<tr><td>✓ Call by reference is used to share the same memory location for actual and formal parameters</td><td>✓ Call by value is used to create a temporary copy of the data which is transferred from the actual parameter in the final parameter.</td></tr>
<tr><td>✓ The changes done in the function are reflected back in the calling environment.</td><td>✓ The changes done in the function in formal parameter are not reflected back in the calling environment.</td></tr>
<tr><td>✓ It makes the use of the & sign as the reference operator.</td><td>✓ It does not use & sign</td></tr>
</table>
</td></tr>
<tr><td>8</td><td>What is constructor? What does it do?</td></tr>
<tr><td>Ans.</td><td>A constructor is member method of a class that is called for initializing the object with initial value when an object is created of that class. Name of constructor and class must be same.</td></tr>
<tr><td>9</td><td>Can you think of the benefits of a private class if any? What are they?</td></tr>
<tr><td>Ans.</td><td>Java does not allow top class as private, but only as inner or nested classes. If a inner or nested class is private, then access is restricted to the scope of that outer class. This feature make Java more secure as we can encapsulate complete class inside a class when we need to protect our class from access.</td></tr>
<tr><td>10</td><td>Write a class specifier (along with its constructor) that creates a class student having two data members: rollno and grade and two methods init() and display().
(do not write full definitions of member methods expect for constructor).</td></tr>
<tr><td>Ans.</td><td><pre>public class student
{
      int rollno; char grade;
      public student()
      {
          rollno=1;
          grade='A';</pre></td></tr>
</table>

```
        }
        public void int(){}
        public void display(){}
}
```

11 **Here is a skeleton definition of a class:**

```
class Sample {
   int i; char C;  float f;
     :
}
```

Implement the constructor.

Ans.

```
public Sample()
{
      i=10; C='Z'; f=75.90;
}
```

12 **What condition(s) a method must specify in order to create objects of class?**

Ans. A method must be constructor in order to create objects of class and must follow these conditions -

a. Name of the method must be same as class name.
b. No return type should be mention, not even void.

13 **Constructor methods obey the usual access rules. What does this statement mean?**

Ans. Constructor methods has obey the rule which is they should be public access modifier, so where ever we want to create object of that class can be able to create object as object creation require constructor to initialization process.

14 **How are parameterized constructors different from non-parameterized constructors?**

Ans.

Parameterized constructor	Non-Parameterized constructor
✓ A Constructor which receives parameters.	✓ A constructor which does not receive any parameter.
✓ **Example:** XYZ 01 = new XYZ(2,13.5F);	✓ **Example:** XYZ 01 = new XYZ();

15 **How do we invoke a constructor?**

Ans. Constructors are automatically invoked by the JVM whenever a new object of respective class is created.

16 **How can objects be initialized with desired values at the time of object creation?**

Ans. By using parameterized constructor objects can be initialized with desired values at the time of object creation.

17 **Write a method that takes two char argument and return 0 if both the arguments are equal. The method return -1 if the first argument is smaller than the second and 1 if the second argument is smaller than the first.**

Ans.

18 **When a compiler can automatically generate a constructor if it is not defined then why is it considered that writing constructors for a class is a good practice?**

Ans. It is good practice to write constructor for a class because we have total control over constructor and objects. Apart from this we can make constructor to do jobs which we want to perform at the time of object creation.

19 **List some of the special properties of the constructor methods.**

Ans.

- ✓ You need not code them explicitly. Java will automatically place a default constructor
- ✓ You can pass arguments to the constructor
- ✓ They can return only an object of type of that class
- ✓ They can be made private
- ✓ They would be executed always (Every time a class is instantiated)

20 **What is parameterized constructor? How is it useful?**

Ans. A parameterized constructor is a constructor which receives parameter (variable) when it is invoked. Parameterized constructor is used to provide different values to the distinct objects and to initialize any member variable at the time of object creation.

TYPE C: Long/Practical Answer Questions

1 **How is call-by-value method of method invoking different from call-by-reference method? Give appropriate examples supporting your answer.**

Ans.

Call By Value	Call by reference
✓ Call by value is used to create a temporary copy of the data which is transferred from the actual parameter in the final parameter.	✓ Call by reference is used to share the same memory location for actual and formal parameters
✓ The changes done in the function in formal parameter are not reflected back in the calling environment.	✓ The changes done in the function are reflected back in the calling environment.

✓ It does not use & sign
Example:
`class passval`
`{`
`public static void main (String agrs[])`
`{`
`// local variable declaration:`
`int a = 100;`
`int b = 200;`
`System.out.println(“Before change, value of a : "+a);`
`System.out.println("Before change, value of b : "+b);`
`change(a, b);`
`System.out.println("After change, value of a : "+a);`
`System.out.println("After change, value of b : "+b);`
`}`
`void change(int x, int y)`
`{`
`x = 10; /* change the value of x */`
`y = 20; /* change the value of y */`
`}`
`}`
Value of a and b did not changed after over writing the value of x and y which contain the value of a and b.
	✓ It makes the use of the & sign as the reference operator. **Example**
	`class passref`
	`{`
	`static int a = 100; static int b = 200;`
	`public static void main (String agrs[])`
	`{`
	`passref c=new passref();`
	`System.out.println("Before change, value of a : "+a);`
	`System.out.println("Before change, value of b : "+b);`
	`c.change(c);`
	`System.out.println("After change, value of a : "+a);`
	`System.out.println("After change, value of b : "+b);`
	`}`
	`void change(passref obj)`
	`{`
	`obj.a = 10; /* change the value of x */`
	`obj.b = 20; /* change the value of y */`
	`}`
	`}`
	Value of a and b is changed after over writing the value of x and y which contain the value of a and b.

2 **Write a method that takes an int argument and doubles it. The method does not return a value. Invoke this method through the GUI application.**

Ans.

```
public void method(int arg)
    {
        int d_arg=arg*arg;
        JOptionPane.showMessageDialog(null, "Doubled Value "+ d_arg);
    }
    private void jButton5ActionPerformed(java.awt.event.ActionEvent evt) {
        method(5);
    }
```

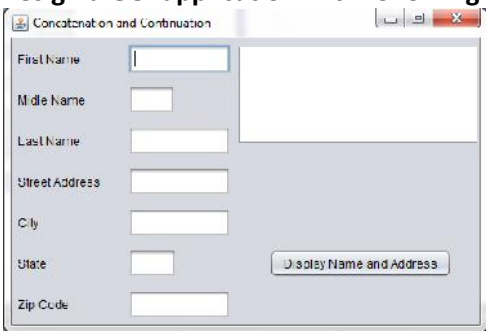
3 **Write a method that takes two char arguments and returns 0 if both the arguments are equal. The method returns –1 if the first argument is smaller than the second and 1 if the second argument is smaller than the first**

Ans.

```
public int method(char ch1,char ch2)
    {
        int x=0;
        if(ch1==ch2)
            x = 0;
        if(ch1>ch2)
```

```
            x = 1;
        if(ch1<ch2)
            x = -1;
        return x;
    }
    private void jButton5ActionPerformed(java.awt.event.ActionEvent evt) {
        System.out.println(method('A','A'));
    }
```

4 **Design a GUI application with following interface:**

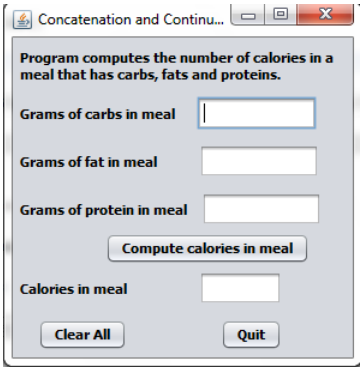


When the user clicks at the push button, all concatenated personal details are shown in the label above the push button. Achieve this task by writing a method namely ShowDetails().

Ans.

```
public void showDetails()
{
  String name=jTextField1.getText() + " "+jTextField2.getText()+"
"+jTextField3.getText();
  String address=jTextField4.getText() + " "+jTextField5.getText()+"
"+jTextField6.getText()+" "+jTextField7.getText();
  jTextArea1.append("name:\t"+name+"\n Address: \t"+address);
}
private void jButton5ActionPerformed(java.awt.event.ActionEvent evt) {
  showDetails();
}
```

5 **Design an application that computes the total calories in a meal, based on the grams of carbohydrates, fats and proteins in the meal.**
A gram of carbohydrates provides 4 calories, a gram of fats provides 9 calories, and a gram of proteins provides 4 calories. Your program must have global variables. (CalperGramInFat, CalperGramInCarb and CalperGramInProtein).
Design a GUI application with following interface:



Your application should look this to start:

(i) Grams are entered into text fields. The total calories are output to a label. The form's labels and button text should be bolded and in size 10 font.

(ii) The user enters the grams of each kind of nutrient, then clicks the Compute calories button. The Compute calories button passes the values (in grams) of each nutrients to a method called CalcTotalCalories().

The method CalcTotalCalories() computes total calories and returns the compute value, which should then be displayed in the label above Quit button. Output should be shown in a different color (e.g. red).

Ans.

```
int CalperGramInFat=9, CalperGramInCarb=4, CalperGramInProtein=4; //GLOBAL
VARIABLES
public void CalcTotalCalories(){
      int
carbCalinMeal=CalperGramInCarb*Integer.parseInt(jTextField1.getText());
      int
fatCalinMeal=CalperGramInFat*Integer.parseInt(jTextField2.getText());
      int
```

```
protCalinMeal=CalperGramInProtein*Integer.parseInt(jTextField3.getText());

jLabel5.setText(Integer.toString(carbCalinMeal+fatCalinMeal+protCalinMeal));
}
private void CalculateActionPerformed(java.awt.event.ActionEvent evt) {
       CalcTotalCalories();
}
private void ResetActionPerformed(java.awt.event.ActionEvent evt) {
        jTextField1.setText(null);
        jTextField2.setText(null);
        jTextField3.setText(null);
        jLabel5.setText(null);
}
private void QuitActionPerformed(java.awt.event.ActionEvent evt) {
        System.exit(0);
}
```

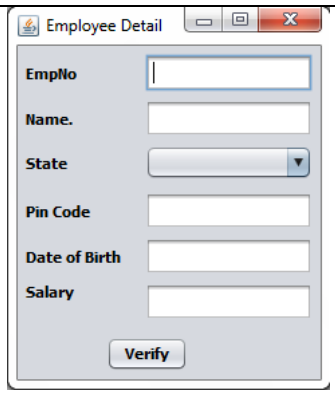
6 **Write a method to receive a long int value and determines whether it has 6 digits or not. Create an application to implement this method.**

Ans.

```
public void CalcTotalCalories(long d)
{
   int len = String.valueOf(d).length();
   if(len<6 || len>6)
      jLabel5.setText("Not 6 Digits ");
   else
      jLabel5.setText("Yes 6 Digits ");
}
private void jButton5ActionPerformed(java.awt.event.ActionEvent evt) {
    long digit=925366L;
    CalcTotalCalories(digit);
}
```

7 **Design a GUI application that obtain all the employee details such as EmpNo (EnoTF), Name (NameTF), State (Statecbox), Pin (PinTF), etc. Name this frme as "EmployeeForm". Add a Birth Date text field named BirthDateTF to the EmployeeForm. Add a Salary text field named SalaryTF to the form. Create a method in the Employee information form called FormCheck() that returns a Boolean value. This method should validate the following information for the fields on the form:**

(i) The nameTF cannot be empty
(ii) The StateCBox combo box must have a state selected.
(iii) The Pin code must be filled in.
(iv) The salary must be > 5000



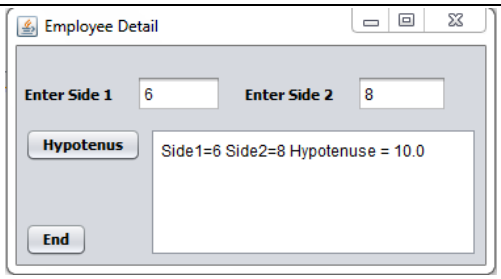
Ans. Change the name of all jTextFields respectively by right clicking each of them then click change variable name and enter the respective name as given in question.

```
public boolean FormCheck()
{
      boolean reply=true;
      String n=NameTF.getText();
      String s=StateCbox.getSelectedItem().toString();
      String p=PinTF.getText();
      int sal=Integer.parseInt(SalaryTF.getText());
      if(n.equals(""))
          reply=false;
      if(s.equals("Select State"))
          reply=false;
      if(p.equals(""))
```

```
            reply=false;
        if(sal<=5000)
            reply=false;
        return reply;
}
private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {
        System.out.println(FormCheck());
}
```

8

Design an application that obtains two sides of a right angle triangle and invokes a method HYPCAL() which calculates the hypotenuse of the triangle using formula. (Use Math.sqrt() inside HYPCAL() method to calculate square root). Your interface should be like the one shown below. Form contains an uneditable textarea where results are displayed.)



Ans.

Uncheck the editable property of jTextArea to make it uneditable.

```
public void HYPCAL()
{
  int side1=Integer.parseInt(jTextField2.getText());
  int side2=Integer.parseInt(jTextField3.getText());
  double h=Math.sqrt((side1*side1)+(side2*side2));
  jTextArea1.append("Side1="+ Integer.toString(side1)+" Side2="+
Integer.toString(side2)+" Hypotenuse = "+Double.toString(h)+"\n");
}
private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {
  HYPCAL();
}
```

9

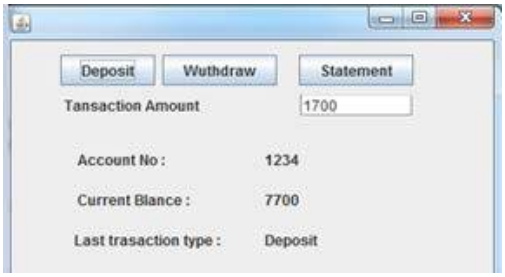
Design a class to represent a bank account. Include the following members:

Data members

- **Name of the depositor**
- **Account number**
- **Type of account**
- **Balance amount in the account**

Design a GUI application having following interface:

Add functionality to above application by creating required methods and implementing them.



Ans.

```
public class NewJFrame1 extends javax.swing.JFrame {
    String name="ABC";
    int number=1234;
    String type;
    int bal=6000;
    int deposit(int bal,int tr_am)
{
    int depo;

    depo=bal+tr_am;
    return depo;
}
    int withdraw(int bal,int tr_am)
{
    int wd;

    wd=bal-tr_am;
    return wd;
```

```
}
//Deposite
    private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {
         int a1;
        a1=Integer.parseInt(jTextField1.getText());
        int sum;
        type="Deposit";
        sum=deposit(bal,a1);
        jLabel6.setText(Integer.toString(sum));
        jLabel5.setText(Integer.toString(number));
        jLabel7.setText(type);
     }
//Withdraw
    private void jButton2ActionPerformed(java.awt.event.ActionEvent evt) {
         int a1;
        a1=Integer.parseInt(jTextField1.getText());
        int w;
        type="withdraw";
        w=withdraw(bal,a1);
        jLabel6.setText(Integer.toString(w));
        jLabel5.setText(Integer.toString(number));
        jLabel7.setText(type);
    }
 //Statement
private void jButton3ActionPerformed(java.awt.event.ActionEvent evt) {
        // TODO add your handling code here:
        jLabel6.setText(Integer.toString(bal));
        jLabel5.setText(Integer.toString(number));
       }
}
```