Q.No.

J

Question Paper Code: D1052

Date of Exam (DDMMYY): 19 06 19

Alpha Numeric Code: BR-CS-D9XE

APJ ABDUL KALAM TECHNOLOGICAL UNIVERSITY
ANSWER BOOKLET

Packet Code. :	DUP-248	Packet Serial No. :	7

Instructions to Candidates:

1. Do not write your Name or Register Number anywhere in this Answer Booklet.
2. Peel the Barcodes corresponding to your Register Number and paste one on the Attendance Sheet and other three on the Answer Booklet - two numbers on the space provided in the facing sheet and one on the last page.
3. Affix your signature on the space provided in the Attendance Sheet.
4. Do not tamper or write on the barcodes as it will mutilate the code. The action will lead to debarring from examinations and withholding the results.
5. Please confine the answers within the space provided in the Answer Booklet. **No additional sheets will be supplied.**
6. DO NOT WRITE ON THE REVERSE OF THIS PAGE.

Name of Exam: B-Tech Semester 4 Degree Examination, May 2019

Course Code: CS206

Course (Subject) Name: OBJECT ORIENTED DESIGN & PROGRAMMING

Branch/Stream: COMPUTER SCIENCE & ENGG

(To be used for first valuation only)

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
a	1	1	2	1		3	3	1	2	1	2	3	3		3	2		
b						2	1					4	3		4			
c																		
d																		
Sub Total	1	1	2	1		5	4	1	2	1	2	7	6		7	2		

	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36
a	4	2																
b		2																
c																		
d																		
Sub Total	4	4																

Marks in words: FIVE-ZERO ~~Fifty~~

Total Marks: 50

(Not to be used for first valuation)

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
a																		
b																		
c																		
d																		
Sub Total																		
	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36
a																		
b																		
c																		
d																		
Sub Total																		

Marks in words | Total Marks

Q.No.

PART - E

15.(a) * The event delegation model defines standard and consistant mechanisms to generate and process events.

* Concept
  - There exists sources which triggers certain events which can be processed by a listener.
  - A listener must simply wait till it receives an event notification.
  - When the listener receives the event, it process the event & returns.

* The listeners have to register with a source inorder to obtain event notification.
* A listener will get event notification only if it wants to, i.e. if it is registered.
* Here, an event source is an object that triggers an event, like a button click, choice selection, mouse movement etc, the event is the change of state in the source due to.
* The event delegation model is more efficient as it cleanly separates the application logic that processes the events and the user interface logic that generates the events.

(b)(i). charAt() - This function is used to extract a character from a specified position.

Q.No.

Syntax: char charAt(int where)

* Here, 'where' is used to specify the index of the character which needs to be extracted.

(II) getChars() — Used to obtain a sequence of characters from a string.

Syntax: String getChars(int sourceStart, int sourceEnd, char target[], int targetStart)

sourceStart — Start index of source string
sourceEnd — Ending index(+1) of source string
target — target array for storing
targetStart — starting index for target array.

(III) equals() — Used to check if two strings are the same.

Syntax: Boolean equals(String str)

* Here 'str' is the name of the string which has to be checked with the calling string.

(IV) replace() — Used to replace a character or character sequence.

Syntax: String replace(OriginalChar, ReplacementChar)
or
String replace(OriginalCharSequence, ReplacementCharSequence)

Q.No.

19.(a) Establishing a JDBC connection involves 5 steps:-

(I) Registering the drivers class.

* Can be done using the function forName() which is used by DriverManager class.

Syntax: static void forName(String className) throws classNotFoundException

eg. DriverManager.forName("jdbc:oracle:database:odbc")

(II) Creating a connection object

* This can be done using the getConnection().

Syntax: static Connection getConnection(String url, String user, String password) throws sqlException

eg. Connection con = getConnection("jdbc/oracle/localhost @1021", myconn, password)

* Last 2 parameters are not necessary.

(III) Creating a statement object

* This can be achieved using the createStatement() which uses Connection interface.

Syntax: Statement createStatement() throws sqlException

eg. Statement stmt = con.createStatement()

Q.No.

(iv) Execute a query

* Can be done using the function executeQuery(), which uses Statement interface.

Syntax: ResultSet executeQuery(String sql) throws sqlException

eg. ResultSet rs = stmt.executeQuery("select * from emp");

(v) Close the connection

* Closing the connection automatically closes the statement & query.
* Can be done using close().

Syntax: void close() throws sqlException

eg. con.close()

(6) * A java applet is very different from other java applications.
* Unlike other java applications, an applet can be used to embed java code into a webpage.
* Unlike other java applications, an applet execute in the web browser & works at client side.
* Other java applications are run as a java class, but applets can only be run via HTML document by including the APPLET tag.
* For other java applications, output is viewed in terminal but applet output is viewed in a

Q.No.

web browser or an applet viewer.

20.6 * paint() method is an AWT method, also used in applets to implement graphical user interface.

* Syntax:

```
paint(Graphics g)
{
    ----
}
```

* It accepts a variable of type graphics.
* It helps to include various functions like drawString() to display strings.
* It is invoked by an event when ever update occurs, paint().

(6) Button

The source generates events when a button is clicked.

* It creates events actionEvent & mouseClickEvent.
* The listeners are actionListener & mouseClickListener.

eg.

```
init()
{
    Button btn = new Button();
    btn.actionListener();
    add(btn);
}
```

Q.No.

```
paint(g){
actionPerformed()
{
    g.drawString("pressed", 10, 50);
}
}
```

(b) Text Components

This source generates events when a text is inputted.

* Events generated are textEvent, keyEvent.
* Listeners are textListener & keyListener.

eg.
```
init()
{
  Text t1 = new Text();
  Text t2 = new Text();
  add(t1);
  add(t2);
  t1.textListener();
  t2.textListener();
}
paint(g)
{
  g.drawString(t1, 10, 50);
  g.drawString(t2, 20, 50);
}
```


Q.No.

PART - D

12.(a) There exists 4 access specifiers in Java, they are :-

(i) Private

A private variable or method can only be accessed from inside the class in which they are declared. ie. It will show an error if they are tried to access outside. Private variable and methods are not inherited.

(ii) Default

This access specifier is the one that is applied if no other access specifiers are used ("default"). Default methods and variables are accessible from the same class or the same package.

(iii) Protected

A protected variable or method can be accessed from inside the same class, same package or by a sub-class.

(iv) Public

A public variable or method can be accessed from anywhere in the program ie. Does not matter if it is different package & not sub-class.

(b) Inheritance is the property by which child classes can use the variables and methods declared in a parent class.

Q.No.

* It is emplemented using the keyword 'Extends'

eg
```
Class A
{
  int i;
  int j;
  void print(){
    system.out.println("i+" "+j);
  }
}

Class B extends A
{
  int k;
  void printK(){
    system.out.println(i+" "+j+" "+k);
  }
}
```

* Here, i & j are also available in class B.
* Private variables and methods cannot be inherited.
* Java supports multiple inheritance ie. Inheritance from 2 parent classes.

* The 'super' keyword is used for referring to the parent class. It can be used for:-

(i) Call parent constructor.

eg
```
Class A{
  int i;
  int j;
  A(){
    i=0;
    j=0;
  }
}

Class B extends A{
  int k;
  B(){
    super();
    k=0;
  }
}
```

Here, super() function is used to call the constructor A() to initialize i & j.

(ii) Access variables & methods of parent class

eg
```
Class A{
  int i;
  int j;
}
Class B extends A{
  int i; int j; int k;
  void sum(){
    system.out.println("Sum:" + (super.i + super.j + k));
  }
}
```

Here, 'super' keyword is used to add k with i & j values of class A.

Q.No.

13. (a) There are 2 streams in java, bytestream and characterstream.

Bytestream

* This stream is used for the input and output of bytes.
* It has 2 abstract classes which defines its base structure, InputStream & OutputStream
* It also has various concreate classes which inherit these abstract classes like ByteArrayInputStream, ByteArrayOutputStream, FileInputStream, FileOutputstream etc.

Characterstream

* This stream is used for the input & output in the form of characters. (Uses UNICODE)
* It has 2 abstract classes which define its base structure, Reader & Writter.
* It has many concrete classes under these abstract classes like FileReader, FileWriter, BufferReader, BufferWriter etc.

(b)
```
Class Thread1 implements Runnable{
    Thread t1;
    Thread1(){
        t1 = new Thread(this, "odd");
        t1.start();
    }
```


Q.No.

```
    public void run(){
        try{
            for(int i=3; i<100; i+=2){
                System.out.println(i);
                t1.sleep(50);}
            }
        catch(InterruptedException e){
            System.out.println("Odd Thread Interrupted");
        }
    }
}
Class Thread2 implements Runnable{
    Thread t2;
    Thread2(){
        t2 = new Thread(this, "even");
        t2.start();
    }
    public void run(){
        try{
            for(int i=2; i<=100; i+=2){
                System.out.println(i);
            }
            t2.sleep(5000);
        }
        catch(InterruptedException e){
            System.out.println("Even Thread Interrupted");
        }
    }
```

Q.No.

```
class DemoThread{
  public static void main(String args[]){
    new Thread1();
    new Thread2();
    try{
      sleep(10000);
    }
    catch(InterruptedException e){
      System.out.println("Main Thread Interrupted");
    }
  }
}
```

PART-C

8. Packages are objects that encapsulate various classes, interfaces and sub-packages. It is used to group certain classes together.
* Packages are created via:-
  Syntax: package pkg;
  Here pkg is the package name.
* Multiple packages are created via:-
  Syntax: package pkg1[.pkg2[.pkg3]];
  Here pkg1, pkg2 & pkg3 are package names.

9. * Interfaces are entities which are used to describe the general layout so that it can be implemented by other classes.
* Interfaces allow only abstract methods, i.e. they can only be used if they are implemented.

Q.No.

* Interface is created using the keyword Interface.

eg.
```
Interface A{
  final int a;
  final int b;
  void show();
}
Class B implements A{
  int c;
  void show(){
    System.out.println(c);
  }
}
```

* Interfaces allow only final or static variables.

10. * There are 3 types of Exceptions, checked exceptions, unchecked exceptions and errors.
* A checked exception is an exception which is found during the compilation of the program.
eg. class NotFoundException, This exception comes up when a specified class is not found by the program.

11. * Thread priority is a number (rank) used to state the priority or importance of a thread.
* A thread with higher priority can prevent another thread with lower priority.
* The priority of a thread can be set with the function setPriority(int P). Here P is assigned a value between MIN_PRIORITY & MAX_PRIORITY.
Can go back to default priority by giving argument

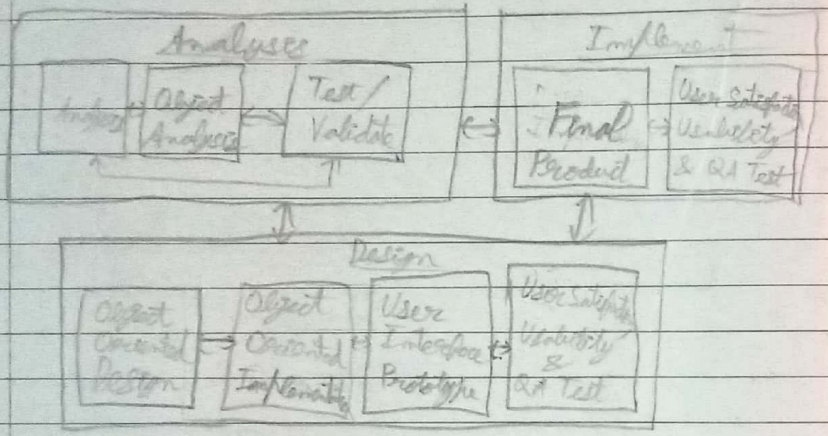
Q.No.

as NORM_PRIORITY.

* Priority of a thread can be obtained using getPriority().

PART – B

6.(a) Object Oriented System Development Life Cycle (OOSDLC)



* It consists of 3 phases :-
Object Oriented Analysis, Object Oriented Design & Object Oriented Testing/Implementation.
* First, it is analysed, the data structures & algorithm required is finalised.
* In 2nd stage, it is designed to perfection.
* In 3rd stage, it is tested & checked how close the product is to the user requirements.

Q.No.

(b)
```
Class Pali{
    public static void main(String args[]){
        int a = Integer.parseInt(args[1]);
        int b = Integer.parseInt(args[2]);
        int a, b;
        a = Integer.parseInt(args[1]);
        b = Integer.parseInt(args[2]);
        for(int i=a; i<b; i++)
        System.out.println("The palindromes between "+ a
        + " and " + b + " are:-\n")
        for(int i=a+1; i<b; i++)
        {
            if(i/100 == i%10)
                System.out.println(i);
            System.out.println("\n");
        }
    }
}
```

7.(a) Method overloading is the concept of the existence of multiple methods with the same name but different number of parameters or datatypes.

eg.
```
Class A {
    int i;
    int j;
    int k;
```

Q.No.

```
I   void sum(int a, int b){
        system.out.println("Sum: "+(a+b));
    }

II  void sum(int a, int b, int c){
        system.out.println("Sum: "+(a+b+c));
    }
    void sum(double a, double b){
        system.out.println("Sum: "+(a+b));
    }
}

class B{
    public static void main(String args[]){
        int a=5;
        int b=4;
        int c=5;
        int d=2.5; e=6.9;
        A ob = new A();
        ob.sum(a,b);   → (I)
        ob.sum(a,b,c); → (II)
        ob.sum(d,e);   → (III)
    }
}
```

Here (I) calls I, (II) calls II & (III) calls III.

(b)

```
Class Queue{
    int front;
    int rear;
    int[] ar = new ar[10];
```


Q.No.

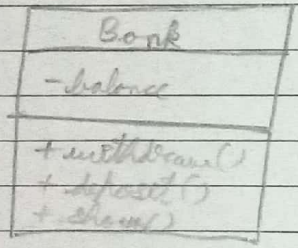
```
    en-queue(int x)
    {
        ar[rear] = x;
        rear++;
    }

    deque{ }
    {
        println(ar[front]);
        ar[front]--;
    }

    display{
        for(i<front; i<rear; i++)
            System.out.println(ar[i]);
    }
}
```

Q.No.

PART-A

1. Polymorphism is the concept that a message can be interpreted by different objects in different ways. Implemented by method overloading.
Encapsulation is the wrapping up of data & code together as a single entity called object.

2. 3 Entities :→ withdraw() → To withdraw cash from balance
(b) deposit() → to deposit cash
(c) show() → To display/check balance



3. Objects are passed as function parameter using the syntax:→
function_name(Object Object_name);

eg.
```
class A{
    int i;
    int j;
    A(){
      i=0; j=0;
    }
}
```

```
void copy(A a)
{
    a.i = i;
    a.j = j;
}
```


Q.No.

```
class B{
    public static void main(String args[]){
        A ob = new A();
        A ob2 = new A();
        ob.copy(ob2);
    }
}
```

Here value of ob2 is same as ob.

4. Parameterised constructors are constructors with parameters or arguments.
A Parameterised constructor can be defined without defining parameter less constructor.
But if parameterised constructor is defined, it cannot call parameter less or even default constructor.

PART-B

16(a) Parameters can be passed to applet using <PARAM> parameter tag.

eg. <APPLET code='Applet File', height=10, width=50>
<PARAM name=x1, value=1>
<PARAM name=x2, value=2>
</APPLET>

Q.No.

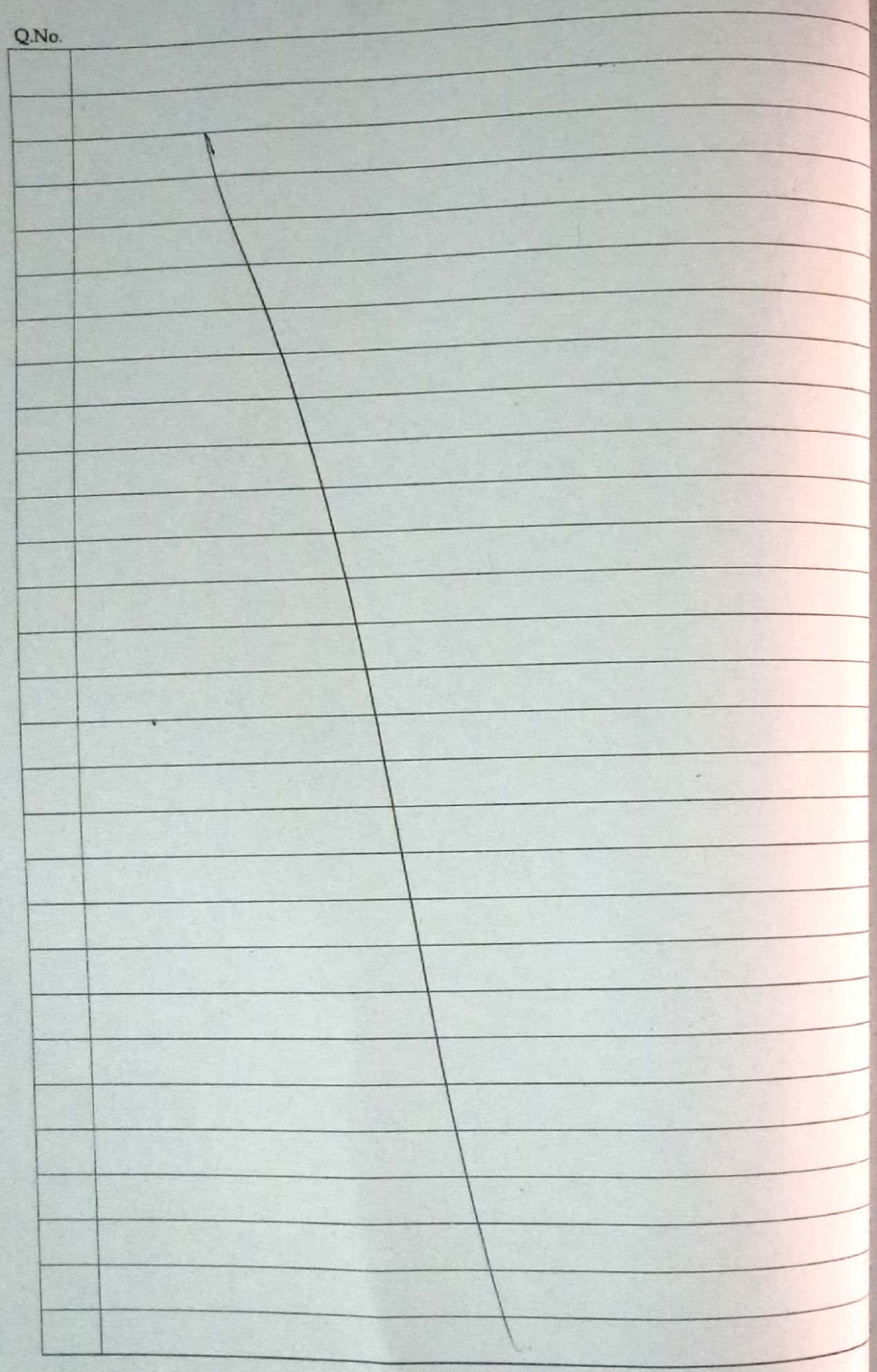
```
paint(){  x= getParameter(x1), y= getParameter(x2),
    a = Integer.parseInt(x1);
    b = Integer.parseInt(x2);
    g.drawString(x1+y, 10, 20);
    g.drawString(x2, 10, 30);
}
```


Q.No.

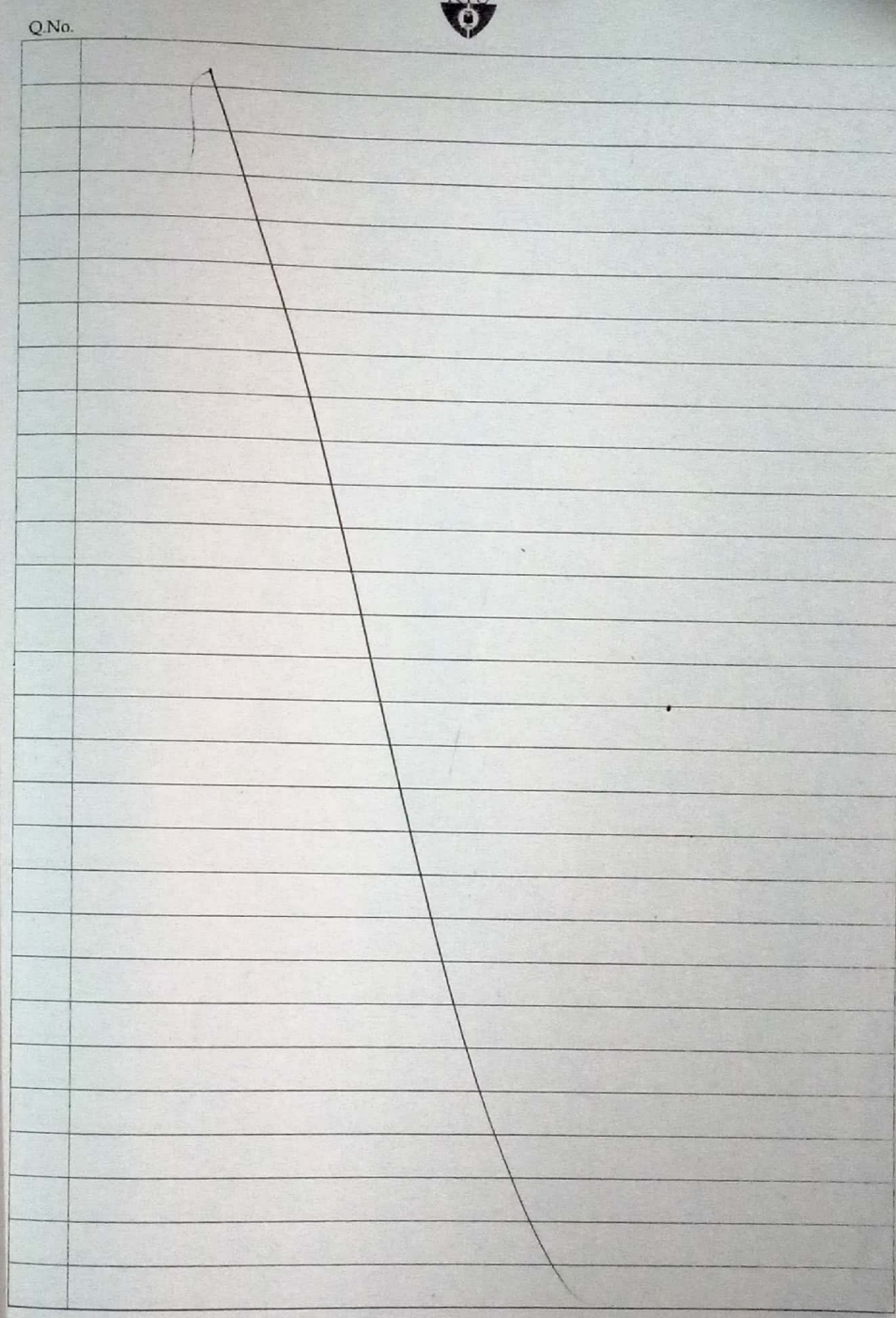
Q.No.

Q.No.

Q.No.



Q.No.



Q.No.

Q.No.

Q.No.

28

Q.No.

29