**Mobile phones are banned**

**M S Ramaiah Institute of Technology**

**Department of Information Science and Engineering**

|  |  |  |  |
| --- | --- | --- | --- |
| **Term:** | 28/07/2017 to 20/12/2017 | **Course Code:** | IS35B |
| **Course:** | Object Oriented Programming with Java | **Semester:** | 3 |
| **CIE:** | Test I | **Max Marks:** | 30 |
| **Date:** | **27/08/17** | **Time:** | **9 :30 –10:30 AM** |

**Portions for Test:** Lecture Nos. from 1 to 17 as per lesson plan

**Instructions to Candidates:** Answer any **Two** out of Three questions.

|  |  |  |
| --- | --- | --- |
| **Sl No** | **Questions** | **Marks** |
| **1a.** | class JavaProgram1  {      public static void main(String[] arg)      {          {   int[] value = { 125, 132, 95, 116, 110 };              int ret\_val = evaluate(value);              System.out.println("The highest score is " + ret\_val);          }      }      public static int evaluate(int[] numbers)      {   int SoFar = numbers[0];           for (int num : numbers)          {              if (num > SoFar)              {                  SoFar = num;              }          }      return SoFar;      }  }  What is the output of the above program? Elaborate the working of the for loop in the program.  Output : The highest score is 132  Working of the for loop | **1 M**  **2M** |
| **1b.** | Design a class called **Author**. Include variables- name (String) ,  email (String) , and gender (char of either 'm' or 'f' - you might also use a boolean variable called isMale having value of true or false). Include a parameterized constructor to initialize the name, email and gender with the given values. Also define functions to return the name, email and gender i.e.- getName(), getEmail(), and getGender(). Class also defines a print () function to print the details of the Author object. Write a main program to show the usage of the Author class.  class Author  {  String name, email;  char gender; // can be of Boolean type isMale() = true (for male) , false (female)  Author(String n, String e, char g)  {  name = n;  email = e;  gender = g;  }  String getName()  {  return name;  }  String getEmail()  {  return email;  }  char getGender()  {  return gender;  }  void print()  {  System.out.println(name + “ “ +email+” “+gender );  }  }  public class MainClass  {  public static void main(String args[])  {  Author a = new Author(“Abc”, “[xyz@gmail.com](mailto:xyz@gmail.com)”, ‘f’);  a.getName();  a.getEmail();  a.getGender();  a.print();  }  } | **4 M**  **3 M** |
| **1c.** | Illustrate method overloading with an example.  Explain Method overloading  Example | **2 M**  **3 M** |
| **2a.** | class Output {     public static void main(String [] args)      {  int I = 0;         outer:             while (true)              { I++;                 inner:                     for (int i = 0; i < 10; i++)                      {                         I += i;                         if (i == 1 || i == 2 || i == 3)                             continue inner;                         break outer;                     }                     continue outer;             }             System.out.println("I = " + I);     } }  What is the output of the above program? Elaborate the role of break and continue in the given program.  I = 1  role of break and continue as goto | **1 M**  **3 M** |
| **2b.** | Define a bank account class with data members name, cust\_id, Balance and account\_number. Use constructor to initialize the values to the members. Define methods to deposit and withdraw money from the account. Write a main method to accept `N’ account details and display those with Balance > 5000.  class Account  {  String name, cust\_id;  float balance, acct\_no;  Account(String n, String c, float bal, float acc\_no)  {  name = n;  cust\_id = c;  balance = bal;  acct\_no = acc\_no;  }  void deposit(float amount)  {  balance+=amount;  }  void withdraw(float amt)  {  balance-=amt;  }  }  public class MainClass  {  public static void main(String args[])  {  int n= 10;  for(int i=0;i<10;i++)  {  //read name, cust\_id, balance,acct\_no  Account ac = new Account(name,cust\_id,balance,acct\_no);  //read amt  ac.deposit(amt);  ac.withdraw(amt);  if(ac.balance>5000)  System.out.println(ac.name+ “ has balance greater than 5000”);  }  }  } | **4 M**  **3 M** |
| **2c.** | Illustrate passing objects as parameters with an example.  passing objects as parameters – call by reference – explain  example | **1**  **3** |
| **3a.** | Define i. Class ii. Object iii. Constructors iv. Finalize()  Class - A *class* is the template from which individual objects are created.  Object – An object is an instance of a class  constructors - **Constructor in java** is a special type of method that is used to initialize the object. Java constructor is invoked at the time of object creation. It constructs the values i.e. provides data for the object that is why it is known as constructor.  Finalize() - The **java**.lang.Object.**finalize()** is called by the garbage collector on an object when garbage collection determines that there are no more references to the object. A subclass overrides the **finalize method** to dispose of system resources or to perform other cleanup. | **4 X 1 = 4 M** |
| **3b.** | Design a Stack class that implements a stack for ‘n’ integers.  public class MyStack {  int maxSize;  int[] stackArray;  int top;    public MyStack(int s) {  maxSize = s;  stackArray = new long[maxSize];  top = -1;  }  public void push(long j) {  stackArray[++top] = j;  }  public long pop() {  return stackArray[top--];  }    public boolean isEmpty() {  return (top == -1);  }  public boolean isFull() {  return (top == maxSize - 1);  }  public static void main(String[] args) {  MyStack theStack = new MyStack(10);  theStack.push(10);  theStack.push(20);  theStack.push(30);  theStack.push(40);  theStack.push(50);    while (!theStack.isEmpty()) {  long value = theStack.pop();  System.out.print(value);  System.out.print(" ");  }  System.out.println("");  }  } | **1**  **1**  **1**  **1**  **2** |
| **3c.** | Write a java program to create a class ClassCompareEqual. Include a constructor to initialize value of 2 data members ‘x’ and ‘y’. Define a compareWith() function to compare two objects and return ‘true’ if greater than else ‘false’ . Also write a main program to test the above class.  class ClassCompareEqual  {  int x,y;  boolean compareWith( ClassCompareEqual ob1)  {  return(x>ob1.x && y>ob1.y);  }  }  class MainClass  {  public static void main(String args[])  {  ClassCompareEqual ob = new ClassCompareEqual();  ClassCompareEqual ob1 = new ClassCompareEqual();  boolean b1 = ob.compareWith(ob1);  }  } | **3 M**  **2 M** |

R- Remember, U-Understand , Ap- Apply and An-analyze.