|  |  |  |  |
| --- | --- | --- | --- |
| **Class: XII Session: 2022 – 23**  **Computer Science (083)**  **Sample Question Paper (Theory)**  **Maximum Marks: 70 Time Allowed: 3 hours**  **General Instructions:**   1. This question paper contains five sections, Section A to E. 2. All questions are compulsory. 3. Section A has 18 questions carrying one mark each. 4. Section B has 7 very short answer type questions carrying 2 marks each 5. Section C has 5 short answer type questions carrying 3 marks each. 6. Section D has 3 long answer type questions carrying 5 marks each. 7. Section E has 2 questions carrying 4 marks each. 8. All programming questions are to be answered using Python Programming Language only. | | | |
| **Section A** | | | |
| **Question No.** |  | **Marks Allocated** |
| **1** | **Name the python object which represents an error.** | **1** |
| **Answer** | *Exception* | |
| **2** | **When do you get syntax errors?** | **1** |
| **Answer** | *When rules of the particular programming language are not followed* | |
| **3** | **Given the list, L1 = [0,1,2,3,4,5,6,7,8,9,10], write the output of the following statement.**  **print(L1[:-1])** | **1** |
| **Answer** | *[0, 1, 2, 3, 4, 5, 6, 7, 8, 9]* | |
| **4** | **Write the output of the following code:**  **print("Python" > "python")** | **1** |
| **Answer** | *False* | |
| **5** | **Write full form of DBMS.** | **1** |
| **Answer** | *DATABASE MANAGEMENT SYSTEM* | |
| **6** | **Which of the following is a valid identifier in Python?**  **a) \_ b) 798 c) for d) a\*\*2** | **1** |
| **Answer** | *only a) \_* | |
| **7** | **Write the output of the following code:**  **my\_dictionary= {1: "John", 1: "Benny", 1: "Abbot", 2: "Sameul"}**  **print(my\_dictionary)** | **1** |
| **Answer** | *{1: 'Abbot', 2: 'Sameul'}* | |
| **8** | **To save any object structure along with the data, we use the module provided by Python. What is the name of the module?** | **1** |
| **Answer** | *Pickle* | |

|  |  |  |
| --- | --- | --- |
| **9** | **How do you add "A" as the first element in the list, my\_list = ["B", "C", "D", "E", "F"]?** | **1** |
| **Answer** | *my\_list.insert(0, "A")* | |
| **10** | **State whether the following statement is True or False.**  **The advantage of using "With" clause is that any file that is opened using this clause is closed automatically.** | **1** |
| **Answer** | *TRUE* | |
| **11** | **Name the Python math built-in function that returns x raised to the power of n.** | **1** |
| **Answer** | *pow(x, n)* | |
| **12** | **Name the protocol which establishes a dedicated and direct connection between two communicating devices.** | **1** |
| **Answer** | *Point to Point Protocol* | |
| **13** | **There are two basic types of firewalls. While one is the Network firewall, name the other firewall.** | **1** |
| **Answer** | *Host-Based Firewall* | |
| **14** | **Give the output of the following code:**  **email\_list = ['aaron.w@gmail.com', 'benny.w@hotmail.com', 'charles.W@gmail.com', 'benny.w@hotmail.com',\**  **'aaron.w@gmail.com', 'benny.w@hotmail']**  **email\_list\_set = list(set(email\_list))**  **print(len(email\_list\_set))** | **1** |
| **Answer** | *4* | |
| **15** | **string = "Fruits such as berries, cherries, plumns, grapefruit, peaches, apples, pears and oranges are lower on glycemic index"**  **s = (string.split(',', maxsplit = 4))**  **print(s[-2])** | **1** |
| **Answer** | *grapefruit* | |
| **16** | **Which of the following mode in file opening statement create a new file if it does not exist or truncates the file if it exists?**  **a) a b) w c) r d) None of these** | **1** |
| **Answer** | *b) w* | |
| **17** | **Fill in the blank.**  **The \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ method is used to write multiple strings to a file. We need to pass an iterable objects such as tuples, lists etc. containing the strings to this method.** | **1** |
| **Answer** | *writelines()* | |
| **18** | **Write the output of the following code:**  **print(162/ 3 + (1+1)\*\*3)** | **1** |
| **Answer** | *62* | |

|  |  |  |  |
| --- | --- | --- | --- |
| **Section B** | | | |
| **19** | **Write the output of the following code:**  **my\_dict = {'Name' : 'Anand' , 'Salary' : 50000.00}**  **my\_dict['Age'] = 48**  **my\_dict['Address'] = "Chennai - 600008"**  **print(my\_dict.keys())**  **print(my\_dict.values())**  **print(my\_dict.items())** | **2** |
| **Answer** | *dict\_keys(['Name', 'Salary', 'Age', 'Address'])*  *dict\_values(['Anand', 50000.0, 48, 'Chennai - 600008'])*  *dict\_items([('Name', 'Anand'), ('Salary', 50000.0), ('Age', 48), ('Address', 'Chennai - 600008')])* | |
| **20** | **Reddy has written a code to input an integer (n) for generating upto n Fibonacci terms. Please re-write the following code and underline the corrections made.**  **print('Program to generate the Fibonacci sequence up to nth term')**  **nterms = (input("How many terms?"))**  **### First two terms**  **term1 = 0**  **term2 = 1**  **count = 0**  **### Validate the input**  **if nterms <= 0:**  **print("You have entered %d " % nterms)**  **print("Please enter a positive integer")**  **elif nterms == 1:**  **print("Fibonacci terms upto %d %d" % (nterms, term1))**  **elif nterms == 2:**  **print("Fibonacci terms upto %d %d" % (nterms, term2))**  **### Generate**  **else:**  **print("Fibonacci sequence")**  **while count < nterms:**  **print(term1)**  **termnth = term1 + term2**  **### Update values**  **term1 = term2**  **term2 = termnth**  **count += 1** | **2** |

|  |  |  |
| --- | --- | --- |
| **Answer** | *print('Program to generate the Fibonacci sequence up to nth term')*  *nterms =* ***int****(input("How many terms?"))*  *### First two terms*  *term1 = 0*  *term2 = 1*  *count = 0*  *### Validate the input*  *if nterms <= 0:*  *print("You have entered %d " % nterms)*  *print("Please enter a positive integer")*  *elif nterms == 1:*  *print("Fibonacci terms upto %d %d" % (nterms, term1))*  *elif nterms == 2:*  *print("Fibonacci terms upto %d %d" % (nterms, term2))*  *### Generate*  *else:*  *print("Fibonacci sequence")*  *while count < nterms:*  *print(term1)*  *termnth = term1 + term2*  *### Update values*  *term1 = term2*  *term2 = termnth*  *count += 1* | |
| **21** | **Name the three commonly used guided / wired media for data transmission and explain how they carry the signals.** | **2** |
| **Answer** | *Three commonly used guided / wired media for data transmission are twisted pair, coaxial cable, and fiber optic cable. Twisted pair and coaxial cable carry electric signals whereas the optical fiber cable carries the light signals.* | |
| **22** | **How do Hackers / Crackers opeate? How are "White Hat Hacker", "Black Hat Hacker" and "Grey Hat Hacker" different from each other?** | **2** |
| **Answer** | *Hackers / Crackers find loopholes and vulnerabilities in computer systems or computer networks and gain access to unauthorised information. If a hacker uses its knowledge to find and help in fixing the security flaws in the system, its termed as White Hacker. If hackers use their knowledge unethically to break the law and disrupt security by exploiting the flaws and loopholes in a system, then they are called black hat hackers. The grey hats take system security as a challenge and just hack system for the fun of it.* | |

|  |  |  |
| --- | --- | --- |
| **23** | **a) Define Stack.**  **b) What are the two fundamental operations performed on the stack? Explain these operations.**  **OR**  **a) We shall create the following two tables:**  **CREATE TABLE employee (**  **EMP\_ID int(6),**  **DEPT\_ID int(6),**  **FIRST\_NAME char(30),**  **LAST\_NAME char(30),**  **AGE int(11)**  **SEX char(1),**  **DOJ datetime**  **);**  **CREATE TABLE department (**  **DEPT\_ID int(6),**  **DEPT\_NAME char(30)**  **);**  **Using the NATURAL JOIN write the SQL Query to list the EMP\_ID, DEPT\_ID, FIRST\_NAME, LAST\_NAME, AGE, SEX, DOJ, DEPT\_NAME from the two tables employee and department.** | **2** |
| **Answer** | *a) Stack is an arrangement of elements in a linear order that follows the principle of Last In First Out (LIFO).*  *b) The two fundamental operations performed on the stack are PUSH and POP. PUSH adds a new element at the TOP of the stack. POP operations is used to remove the top most element of the stack.*  *OR*  *a) SELECT \* from employee NATURAL join department;* | |
| **24** | **What is a Primary Key in a Relational Database Management System> Why do you need a Primary Key?** | **2** |
| **Answer** | *Primary Key in a relation is a one or more of the candidate keys chosen to uniquely identify the tuples in that relation.*  *Maintaining data consistency is a critical task and the most data consistency problems is duplicate records. Primary Key helps prevent this.* | |
| **25** | **a) Write the full form of the following:**  **i) MAC ii) OUI**  **b) What is a Protocol in Communication? Give at least three examples.** | **2** |
| **Answer** | *a) (i) MAC stands for Media Access Control*  *(ii) OUI standas for Organisational Unique Indentifier*  *b) In communication, protocol is a set of standard rules that the communicating parties (the sender, the receiver, and all other intermediate devices) need to follow.*  *Flow Control, Access Control and addressing are three examples of protocol.* | |
| **Section C** | | | |
| **26** | **Your client wants a Sales System using MySQL to store the data. As a Database administrator, Anand decided that:**  **To create a sales table which describes the information about each sales for each sales person.**  **Name of the database: mysales**  **Name of the table: SALES**  **Attributes of the SALES are as follows:**  **SALES\_PERSON\_ID - Numeric**  **SALES\_PERSON\_NAME - Character of size 30**  **SALES\_DATE - Date**  **SALES\_REGION - Character of size 15**  **SALES\_AMOUNT - Numeric**  **a) Identify the attribute suitable to be declared as a primary key and justify.**  **b) For the Sales person with ID 1, Modify the SALES\_AMOUNT column from value to 16000.**  **c) Write a query to list all the tables in the database, mysales.** | **3** |
| **Answer** | *a) When choosing a primary key, we need to choose a column that's used to uniquely identify each record in a table. In the SALES table, SALES\_PERSON\_ID uniquely identifies the Sales person and it is chosen and declared as a primary key.*  *b) UPDATE SALES SET SALES\_AMOUNT = 16000 WHERE SALES\_PERSON\_ID = 1;*  *c) show TABLES;* | |
| **27** | **Write the query based on the table given below:**  **Table: SALES**   |  |  |  |  |  | | --- | --- | --- | --- | --- | | SALES\_PERSON\_ID | SALES\_PERSON\_NAME | SALES\_DATE | SALES\_REGION | SALES\_AMOUNT | | 1 | Abel Paul | 2021-07-01 09:50:06 | CHENNAI | 16000 | | 2 | Anand Babu | 2022-07-01 09:54:44 | CHENNAI | 15000 | | 3 | Bhaskar D | 2021-11-02 09:55:35 | COIMBATORE | 20000 | | 4 | Charles David | 2022-04-14 09:56:44 | MADURAI | 25000 | | 5 | Edward Kingsley | 2022-01-19 09:57:29 | TRICHY | 50000 |   **i) Get the regions that are unique from the table, SALES.**  **ii) Give the name of the Sales person whose sales amount is MORE than the average sales amount.**  **iii) Which Sales person made the recent sales?** | **3** |

|  |  |  |
| --- | --- | --- |
| **Answer** | *i) SELECT DISTINCT(SALES\_REGION) FROM SALES;*  *ii) SELECT SALES\_PERSON\_NAME, SALES\_AMOUNT FROM SALES*  *WHERE SALES\_AMOUNT > (SELECT AVG(SALES\_AMOUNT) FROM SALES);*  *iii) SELECT SALES\_PERSON\_NAME, SALES\_DATE FROM SALES WHERE SALES\_DATE = (SELECT MAX(SALES\_DATE) FROM SALES);* | |
| **28** | **Table: COMPANY**   |  |  |  | | --- | --- | --- | | COMPANY\_ID | COMPANY\_NAME | COMPANY\_CIN | | 1 | Reliance Industries | L17110MH1973PLC019786 | | 2 | Oil & Natural Gas | L74899DL1993GOI054155 | | 3 | Indian Oil | L23201MH1959GOI011388 | | 4 | Tata Motors | L28920MH1945PLC004520 | | 5 | Tata Consultancy Services | L22210MH1995PLC084781 |   **I) From the table, COMPANY and the column, COMPANY\_CIN, write SQL query to extract two characters which bear the State Code starting from the 7th position.**  **II) Write SQL query to select records with company name starting with the letter, "T".**  **iii) Write SQL query to select records with CIN containing the characters MH in positions 7 and 8.** | **3** |
| **Answer** | *I) SELECT DISTINCT SUBSTRING(COMPANY\_CIN, 7, 2) FROM COMPANY;*  *ii)SELECT \* FROM COMPANY WHERE COMPANY\_NAME LIKE 'T%';*  *iii) SELECT \* FROM COMPANY WHERE COMPANY\_CIN LIKE '%MH%';* | |
| **29** | **Write a program to create a stack for storing only prime numbers from the list, given below:**  **integer\_List = [ 1, 2, 3, 4, 5, 6, 7, 8, 9, 10].**  **Display the content of the stack.** | **3** |
| **Answer** | *def isPrime(num):*  *if (num < 2):*  *return False*  *for i in range(2, num//2 + 1):*  *if (num % i == 0):*  *return False*  *return True*  *integer\_List = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]*  *stack = []*  *for x in integer\_List:*  *if isPrime(x):*  *stack.append(x)*  *print("Our Stack is {}".format(stack))* | |

|  |  |  |
| --- | --- | --- |
| **30** | **i) Write whether the following statement is True or False.**  **In the Python Programming language, all arguments are supplied by Value.**  **Ii) Write two benefits of using Python Function.**  **OR**  **Write a function to calculate the factorial of a number. Using this function calculate the factorial for the numbers given in the list, fact\_list = [3, 4, 5, 6]** | **3** |
| **Answer** | *i) False. In the Python Programming language, all arguments are supplied by Reference.*  *Ii) Benefits of using Python Function:*  *a) By using functions, we can prevent repeating the same code block repeatedly in a program as we can use python functions with different arguments and return as many outputs as we want.*  *b) A large Python program can be separated into numerous functions to manage the program easily.*  *OR*  *def factorial(x):*  *if x == 1:*  *return 1*  *else:*  *return (x \* factorial(x - 1))*  *numbers\_list = [3, 4, 5, 6]*  *factors\_list = [ ]*  *for num in numbers\_list:*  *fac = factorial(num)*  *factors\_list.append(fac)*  *print("\nFactorial of {} is {} respectively". format(numbers\_list, factors\_list))* | |

|  |  |  |  |
| --- | --- | --- | --- |
| **Section D** | | | |
| **31** | **(a) Write the output of the following Python Code:**  **def split\_into\_rows(name\_list, mod = 3):**  **for index, name in enumerate(name\_list, start = 1):**  **print(f"{name:-^15} ", end = "#")**  **if index % mod == 0: # check if the remainder of a division is zero**  **print()**  **print()**  **names\_list = ['Apple', 'Apricot', 'Avocado', 'Cranberry','Date plum']**  **split\_into\_rows(names\_list)**  **(b) Write the output of the SQL queries given below:**  **Table: EMPLOYEE\_DETAILS**   |  |  |  |  |  | | --- | --- | --- | --- | --- | | **ED\_ID** | **ED\_NAME** | **ED\_SALARY** | **ED\_DOJ** | **DEPT\_ID** | | **1001** | **ABEL PAUL** | **15000** | **2000-10-01** | **NULL** | | **1002** | **ANN MARY** | **20000** | **1998-09-01** | **10** | | **1003** | **BHASKAR CD** | **25000** | **2020-08-01** | **10** | | **1102** | **DAVID ROBERT** | **45000** | **1997-06-01** | **20** | | **1104** | **CHARLES JOHN** | **95000** | **1990-07-01** | **20** |   **i) SELECT DISTINCT DEPT\_ID FROM EMPLOYEE\_DETAILS;**  **ii) SELECT DEPT\_ID, AVG(ED\_SALARY), COUNT(\*) FROM EMPLOYEE\_DETAILS GROUP BY DEPT\_ID;**  **iii) SELECT ED\_NAME, ED\_DOJ FROM EMPLOYEE\_DETAILS**  **WHERE ED\_DOJ = (SELECT MAX(ED\_DOJ) FROM EMPLOYEE\_DETAILS);**  **iv) SELECT AVG(ED\_SALARY) FROM EMPLOYEE\_DETAILS;** | 1. **+ 3** |
| **Answer** | *a) -----Apple----- #----Apricot---- #----Avocado---- #*  *---Cranberry--- #---Date plum--- #*  *b) ( i)*  *DEPT\_ID*  *NULL*  *10*  *20*  *(ii)*   |  |  |  | | --- | --- | --- | | *DEPT\_ID* | *AVG(ED\_SALARY)* | *COUNT(\*)* | | *NULL* | *15000* | *1* | | *10* | *22500* | *2* | | *20* | *70000* | *2* |   *( iii)*   |  |  | | --- | --- | | *ED\_NAME* | *ED\_DOJ* | | *BHASKAR CD* | *2020-08-01* |   *(iv)*  *AVG(ED\_SALARY)*  *40000* | |

|  |  |  |
| --- | --- | --- |
| **32** | 1. **Write a program in Python that defines and calls the following user defined functions:**   1) **APPEND() - To accept and append data of a customer in the CSV file, 'actors.csv'. Each record contains a list of fields such as id, name and city to store the customer\_id, customer\_name and customer\_city respectively.**  **2) COUNTR() - To count the number of records present in the CSV file named 'actors.csv'.**  **b) The code given below inserts the following record in the table, sales\_details.**  **ID of the Sales person – Integer**  **Date of Sale - Date (format yyyy/mm/dd)**  **Sale amount in Rupees - Integer**  **Note the following information is used to establish the connectivity between Python and MySQL.**  **Host: localhost**  **User: userA**  **Password: P1sswor4**  **Database: sales**  **Write the following missing statements:**  **import mysql.connector as mysql**  **def sql\_insert\_data():**  **connection = mysql.connect(\_\_\_\_\_\_\_\_\_\_\_\_\_) # Statement 1**  **cursor = connection.cursor()**  **myID = int(input("Enter SALES PERSON (Numeric) :: "))**  **mySalesDate = input("Enter Sales DATE in yyyy/mm/dd format :: ")**  **mySales = int(input("Enter SALE AMOUNT IN RUPEES :: "))**  **mySql\_insert\_query = """INSERT INTO sales.sales\_details (ID,SALE\_DATE, SALE\_AMOUNT) VALUES (%s, %s, %s);"""**  **recordTuple = \_\_\_\_\_\_\_\_\_\_\_\_ # Statement 2**  **cursor.execute(\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_) # Statement 3**  **connection.commit()**  **print("Sales Data for person %d data added successfully " % myID)**  **connection.close()**    **sql\_insert\_data()** | **3 + 2** |

|  |  |
| --- | --- |
| **Answer** | *a(1)*  *def APPEND(row):*  *import csv*    *try:*  *with open('actors.csv', 'a', newline = '') as csvFile:*  *writer = csv.writer(csvFile)*  *writer.writerow(row)*  *except:*  *print("\nError in appending\n")*  *finally:*  *print("Appended the row successfully")*    *row = ['7', 'M G Ramachandran', 'Chennai']*  *APPEND(row)*  *row = ['8', 'Sivaji Ganesan', 'Chennai']*  *APPEND(row)*  *print("\nAdded rows to the csv file\n")*  *2)*  *def COUNTR():*  *with open('actors.csv', 'r') as fp:*  *for count, line in enumerate(fp):*  *pass*  *return count + 1*  *print('Total Lines', COUNTR())* |
| *b)*  *import mysql.connector as mysql*  *def sql\_insert\_data():*  ***connection = mysql.connect(host = "localhost", user = "userA", password = "P1sswor4", database = "sales")******# Statement 1***  *cursor = connection.cursor()*  *myID = int(input("Enter SALES PERSON (Numeric) :: "))*  *mySalesDate = input("Enter Sales DATE in yyyy/mm/dd format :: ")*  *mySales = int(input("Enter SALE AMOUNT IN RUPEES :: "))*  *mySql\_insert\_query = """INSERT INTO sales.sales\_details (ID,SALE\_DATE, SALE\_AMOUNT) VALUES (%s, %s, %s);"""*  ***recordTuple = (myID, mySalesDate, mySales)*** *#* ***Statement 2***  ***cursor.execute(mySql\_insert\_query, recordTuple)*** *#* ***Statement 3***  *connection.commit()*  *print("Sales Data for person %d data added successfully " % myID)*  *connection.close()*    *sql\_insert\_data()* |

|  |  |  |
| --- | --- | --- |
| **33** | **(a) Write a function in Python PUSH(Lis), where Lis is a list of strings, ['ABC', 'BCD', 'CDE', 'DEF', 'EFG', 'FGH']. From this list push all strings having no vowels into a stack implemented by using a list.**  **Display the stack if it has at least one element, otherwise display appropriate error message.**   1. **(i) What is Media Control Access (MAC) Address?** 2. **What is Internet Protocol (IP) Address? Explain.** | **3 +2** |
| **Answer** | *(a)*  *def isVowels(string): # use-defined function*  *count = 0*  *vowels\_list = ['a', 'e', 'i', 'o', 'u']*  *string\_lower = string.lower()*    *while count < len(string\_lower):*  *if string\_lower[count] in (vowels\_list):*  *return True*  *count = count+1*  *return False*  *str\_list = ['ABC', 'BCD', 'CDE', 'DEF', 'EFG', 'FGH']*  *stack = []*  *novowel\_count = 0*  *for str in str\_list:*  *if isVowels(str):*  *pass*  *else:*  *stack.append(str)*  *novowel\_count += 1*  *if novowel\_count == 0:*  *print('\nStack is empty - All strings contain letters of vowels')*  *else:*  *print("\nStack {}".format(stack))*  (b)  I) **Media Control Access Address:**  This is also known as the Physical or hardware address and is a unique permanent value associated with a network adapter called a NIC. It is used to physically identify a machine on the network.  ii) **Internet Protocol Address:**  This is known as Internet Protocol address and is a unique address that can be used to uniquely identify each node in a network. Unlike MAC address, IP address can change if a node is removed from one network and connected to another network. | |

|  |  |  |
| --- | --- | --- |
| **Section E** | | |
| **34** | **Aziz creates a table, invoice\_payment with a set of records to track and collect the invoice amount due from the customers.**  **He had entered data for 6 customers in the table.**   |  |  |  | | --- | --- | --- | | CUST\_ID | INVOICE\_AMOUNT | INVOICE\_DATE | | 1 | 50000 | 2022-01-31 | | 2 | 45000 | 2022-04-30 | | 3 | 30000 | 2022-02-28 | | 4 | 25000 | 2022-03-31 | | 5 | 20000 | 2022-06-30 | | 6 | 15000 | 2022-07-31 |     **Based on the given above, write SQL statements to answer the following questions:**   1. **Add a column PAYMENT\_DUE\_DATE to the table, invoice\_payment** 2. **Add 30 days to the INVOICE\_DATE and to fill in the PAYMENT\_DUE\_DATE column and calculate the number of days from the PAYMENT\_DUE\_DATE up to today.** 3. **Calculate INTEREST\_AMOUNT as interest @10% per annum ON the INVOICE\_AMOUNT for the overdue days.**   **Show INVOICE\_AMOUNT, PAYMENT\_DUE\_DATE, OVERDUEDAYS and INTEREST** | **1 + 1 + 2** |
| **Answer** | *ALTER TABLE invoice\_payment*  *ADD PAYMENT\_DUE\_DATE date;*  *UPDATE invoice\_payment*  *SET PAYMENT\_DUE\_DATE = ADDDATE(INVOICE\_DATE,INTERVAL 30 DAY);*  *select INVOICE\_AMOUNT, PAYMENT\_DUE\_DATE, datediff(curdate(),PAYMENT\_DUE\_DATE ) AS OVERDUEDAYS,*  *ROUND((datediff(curdate(),PAYMENT\_DUE\_DATE ) / 365) \* 0.10 \* INVOICE\_AMOUNT,2) AS INTEREST*  *from invoice\_payment;* | |

|  |  |  |
| --- | --- | --- |
| **35** | 1. **The following program accepts a record of salesman from the user and appends it in the binary file, salesman.dat. It prints the count of records appended and also the size of the file created.**   **As a Python expert, complete the following code based on the requirement given above:**  # Write salesman record in a binary file and dumped using pickle  import pickle    print("\nProgram to enter salesman data")    bfile = **\_\_\_\_\_\_\_\_\_\_\_\_\_ # Statement 1**  recno = 1  print("Enter Records of Salesman")  # Get data from the user and dump in the file as list object  while True:  print("\n== Record No == %d" % recno)  s\_id = int(input("\t Sales Person Number : :"))  s\_customer = input("\t Customer Name : :")  s\_date = input("\t Sale Date (YYYY/MM/DD : : ")  s\_amount = int(input("\t Sale Amount : : "))  s\_commission = round(s\_amount \* 0.10, 0)  s\_data = [s\_id, s\_customer, s\_date, s\_amount, s\_commission]    **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ # Statement 2**    ans = input("Do you want to enter more records? (y/n) : :")  if ans.lower() == 'n':  print("\n Entered %d records\n" % recno)  break  recno += 1  print("\nFile size in bytes %d" %bfile.tell())  ### Close the file  bfile.close()   1. **From the binary file, salesman.dat, display all the records with record number.**   ### Retrieve the file from the binary file dumped using Pickle  import pickle  recno = 1  try:  **with \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_: # Statement 3**  while True:  sdata = \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ **# Statement 4**  print("\nRecord No :: %d \n" % recno)  print(sdata)  recno += 1  except EOFError:  pass  bfile.close() | **1 + 1 + 1 + 1** |
| **Answer** | **(a)**  *import pickle*    *print("\nProgram to enter salesman data")*    ***bfile = open("salesman.dat", "ab") # Statement 1***  *recno = 1*  *print("Enter Records of Salesman")*  *# Get data from the user and dump in the file as list object*  *while True:*  *print("\n== Record No == %d" % recno)*  *s\_id = int(input("\t Sales Person Number : :"))*  *s\_customer = input("\t Customer Name : :")*  *s\_date = input("\t Sale Date (YYYY/MM/DD : : ")*  *s\_amount = int(input("\t Sale Amount : : "))*  *s\_commission = round(s\_amount \* 0.10, 0)*  ***s\_data = [s\_id, s\_customer, s\_date, s\_amount, s\_commission] # Statement 2***    *pickle.dump(s\_data, bfile)*    *ans = input("Do you want to enter more records? (y/n) : :")*  *if ans.lower() == 'n':*  *print("\n Entered %d records\n" % recno)*  *break*  *recno += 1*  *print("\nFile size in bytes %d" %bfile.tell())*  *### Close the file*  *bfile.close()* |  |
| **Answer** | *### Retrieve the file from the binary file dumped using Pickle*  *import pickle*  *recno = 1*  *try:*  ***with open("salesman.dat", "rb") as bfile: # Statement 1***  *while True:*  ***sdata = pickle.load(bfile) # Statement 2***  *print("\nRecord No :: %d \n" % recno)*  *print(sdata)*  *recno += 1*  *except EOFError:*  *pass*  *bfile.close()* | |
|  | | |