 **Indian School Al Wadi Al Kabir**

**Assessment – 1**

**COMPUTER SCIENCE (Code: 083)**

CLASS : XII ANSWER KEY Max. Marks:70

|  |  |  |
| --- | --- | --- |
|  | **SECTION A** |  |
| 1. |  False | 1 |
| 2. |  b. 14.0 | 1 |
| 3. | c. user\_2  | 1 |
| 4. |  c. INSERT  | 1 |
| 5. | a. ext#nex | 1 |
| 6. |  b.Binary | 1 |
| 7. |  a. DISTINCT | 1 |
| 8. | c. puC dlroW  | 1 |
| 9. | d. Shuffle 11 # Draw 50 | 1 |
| 10. |  c. BETWEEN | 1 |
| 11. |  b. True  | 1 |
| 12. | a. 20,6  | 1 |
| 13. | d. [‘Economy’,’Digital’] | 1 |
| 14. |  b. 20 | 1 |
| 15. |  d. (5,6) | 1 |
| 16. |  c. ALTER  | 1 |
| 17 |  b. LIKE operator |  |
| 18 |  c. HAVING |  |
| 19 | 1. Removes the first occurrence of value x from the list
 |  |
| 20. |  Assertion (A) : If numeric data are to be written to a text file, the data needs to be converted into a string before writing to the file. Reason (R) : write() method takes a string as an argument and writes it to the text  file**(A) Both Assertion (A) and Reason (R) are true, but Reason (R) is the correct explanation of Assertion (A).**  | 1 |
| 21. |  Assertion ( A) : In SQL, the aggregate function Avg() calculates the average  value on a set of values and produce a single result.  Reason ( R) : The aggregate functions are used to perform some  fundamental arithmetic tasks such as Min(),Max(), Sum() etc   **(B) Both Assertion (A) and Reason (R) are true, but Reason (R) is not the correct explanation of Assertion (A).**   | 1 |
|  | **SECTION B** |  |
| 22 | i. ALTER TABLE SPORTS ADD COLUMN CATEGORY VARCHAR(20); ii. USE EXAM; SHOW TABLES; | 2 |
| 23 | **Twi ink Twi** | 2 |
| 24 |  def search(teacher, tname): for i in range(len(teacher)): if tname.lower()==teacher[i].lower(): print(teacher[i] , ‘at’, i) | 2 |
| 25 |  DEL@COL@BEI | 2 |
| 26 |  (i) lst.remove(10) (ii) str1= str1.replace(“This”, “That’)  | 2 |
| 27. |  DELETE FROM STUDENT  WHERE NAME=’RAHUL’ AND STUD\_ID = 100; INSERT INTO STUDENT VALUES(123,‘RAJEEV’,12,’SCIENCE’);  | 2 |
| 28 |  6$30  | 2 |
|  | **SECTION C** |  |
| 29. | def COUNTWORDS(): c=0 f=open(“DECODE.TXT”,’r’) s=f.read() l=s.split() for word in l: if len(word)>=5: c=c+1 f.close() return c  ORdef COUNTLINES(): f=open(“CONTENT.TXT”,’r’) l=f.readlines() c=0 for line in l: w=line.split() if len(w)>=5: c=c+1 print(c) f.close( ) | 3 |
| 30. |  (i) ALTER TABLE RENT\_CAB ADD PRIMARY KEY(Vcode);(ii) UPDATE RENT\_CAB SET CHARGES=CHARGES +CHARGES \*10/100 ;(iii) ALTER TABLE RENT\_CAB DROP COLUMN COLOR; | 3 |
| 31. |  CITY = [] def push\_city(d\_city): for k in d\_city: if len(k)>4: CITY.append(d\_city[k]) def pop\_city(): while True: if CITY==[]: print(“stack empty”) break else: print(CITY.pop()); | 3 |
|  | **SECTION D** |  |
| 32. |  i. The process of exception handling involves writing additional code to give proper messages or instructions to the user. This prevents the program from crashing abruptly. The additional code is known as an exception handler. ii.  try: x=x+10 except NameError:  print(" Name is not defined") except:  print("Some Error Occurred") | **4** |
| 33 | 1. SELECT AVG(PRICE) FROM STATIONERY

GROUP BY DISTRIBUTOR HAVING SUM(QTY)>=200;1. SELECT \* FROM STATIONERY ORDER BY PRICE DESC;
2. SELECT DISTINCT DISTRIBUTORS FROM STATIONERY;
3. SELECT SUM(PRICE)FROM STATIONERY WHERE QTY<100;
 |  |
| 34. | import csvdef Add\_details(): f=open(“sports.csv”,’a’) w=csv.writer(f) sportid=int(input(“enter id”)) comp=input(“enter competition”) prize=input(“enter prize”) rec=[sportsid,comp,prize] w.writerow(rec) f.close()def count\_medal(): f=open(“sports.csv”,’r’) r=csv.reader(f) for rec in r: if rec[2]==’Gold’: print(rec[1]) f.close() | 4 |
| 35 | (i) What is the main purpose of seek() and tell() method ? seek() tell()Purpose : Repositions the file pointer to a Returns the current position  Specific location within a file of the file pointerSyntax : seek(offset [,reference point]) tell()Parameters: Requires specifying the offset Requires no parameters and an optional reference point (ii)import pickledef search\_copy(): f1= open(“cinema.dat’,’rb’) f2=open(‘movie.dat’,’wb’) try: while True: rec=pickle.load(f1) if rec[2]==’comedy’: pickle.dump(rec.f2) except EOFError: f1.close() f2.close() iii) import csvdef CountGradeAB(Sub) : f=open(“TEST.CSV”,’r’) r=csv.reader(f) c1=0 c2=0 for rec in r: if rec[1]==sub: if int( rec[3])>=90: c1=c1+1 elif int(rec[3])>=80: c2=c2+1 print(“ Number of Grade A :” , c1) print(“ Number of Grade B :”, c2) f.close() | 1+3 |
|  | **SECTION E** |  |
| 36. |  1. SELECT COUNT(\*), AVG(ORDAMT) FROM CLIENT

GROUP BY CTYPE; ii. SELECT \* FROM CLIENT WHERE CTYPE = ‘REGULAR’ ORDER BY ORDAMT ASC; iii. SELECT CNAME,GENDER,ORDDATE FROM CLIENT WHERE  CNAME LIKE ‘%S’;  iv. SELECT \* FROM CLIENT  WHERE CTYPE NOT IN ( ‘REGULAR’, ‘NEW’); v. SELECT CNAME,CTYPE FROM CLIENT  WHERE ORDDATE < ‘1985/05/10’; | 5 |
| 37. | import pickledef appendData(): f=open(‘PRODUCTS.DAT’, ‘ab’) pid=int(input(“ENTER ID:”)) name=input(“enter name:”) price = int(input(“enter price:”)) rec={‘PID’:pid, ’PNAME’:name, ’PRICE’: price} pickle.dump(rec,f) f.close()def findProduct(product\_id): f=open(“PRODUCTS.DAT”,’rb’) try: while True: rec=pickle.load(f) if rec[0]==product\_id: print(rec) except EOFError: f.close()  | 5 |