# PRACTICE PAPER – 1 (SOLUTION)

## CLASS : XII

## SUBJECT: COMPUTER SCIENCE

### SECTION-A

**1.** False
**2.** (B) for statement
**3.** (A) Entry Controlled Loop
**4.** (B) \nhello
**5.** (A) [ ]
**6.** (D) Method
**7.** (C) Required arguments
**8.** (B) Tables
**9.** (C) It returns the byte position of the file pointer as an integer
**10.** (B) Attribute
**11.** (D) All of these
**12.** (C) JOIN
**13.** (B) Message
**14.** (A) "John"
**15.** (B) cursor.callproc('procedurename', [parameters,])
**16.** (B) resultest
**17.** (B) Both A and R are true and R is not the correct explanation for A
**18.** (D) A is false but R is True

### SECTION-B

**19.**
```
c=dict()
n-int(input("Enter total number"))
i=1
while i<=n:
    a=input("enter place")
    b=input("enter number")
    c[a]=b
    i = i + 1
print  ("place","\t","number")
for i in c:
        print (i,"\t",c[i])
```

**20.** (i) Simple Mail Transfer Protocol

(ii) Voice over Internet Protocol (Voice over IP)

(iii) Global System for Mobile Communication

(iv) Wireless Local Loop

**OR**

**Packet switching:**

- Uses store and forward concept to send messages
- No physical path is actually establishes
- Message is divided into smaller parts, known as packets and then sent forward
- Tight upper limit on block size
- Each data unit knows only the final receiver's address

**Circuit switching**

- Physical Connection is established between sender and receiver
- Each data unit knows the entire path from sender to receiver
- It does not follow store and forward concept

**21.** (a) SCHOOLbbbbCOM

(b) 250 # 150

250 # 100

130 # 100

**22.** A relation is a table having atomic values, unique and unordered rows and columns. The relational model represent data and relationship among data by a collection of tables known as relation, each of which has a number of columns with unique names.

**23.** **(a)** Website is a collection of web pages belonging to a particular person or organisation.

**(b)** To see the source code, right click on the web page and then click on the" view source option.

**24.** [ ] & [25,20,15,10]

**OR**

(1,2,[1,3.14],3)

**25.** Equi join is a simple SQL join condition that uses equal sign as a comparison operator.

**Syntax**

SELECT column1, column2, column3 FROM Table1, Table2 WHERE Table1. columnl = Table2. column1

**OR**

DDL: DROP TABLE, ALTER TABLE

DML: INSERT INTO, UPDATE....SET

## SECTION-C

**26.** **(a)**

| COUNT(*) | City |
|---|---|
| 2 | Mumbai |
| 2 | Delhi |
| 2 | Moscow |

**(b)** (i)

| MAX (TRAVELDATE), | MIN (TRAVELDATE) |
|---|---|
| 2018-11-10 | 2018-05-09 |

(ii)

| **End** | **Count(*)** |
|---|---|
| Habibganj | 2 |
| Amritsar Junction | 2 |
| New Delhi | 4 |

**Note :** Values may be written in any order

(iii) **DISTINCT TRAVELDATE**

2018-12-25
2018-11-10
2018-10-12
2018-05-09

**Note :** Values may be written in any order

(iv)

| **TNAME** | **PNAME** |
|---|---|
| Amritsar Mail | N S SINGH |
| Swarna Shatabdi | R SHARMA |

**27.**

```
def CountFirstDigit( ) :
    count = 0
    with open('Diary.txt','r') as f:
    while True:
        line=f.readline( )
            if not line:
            break
            if line[0] .isdigit() :
                count = count+1
    if   count == 0:
        print ("no  line  starts  with a digit")
    else:
        print ("count=",count)
```

**OR**

```
def CountDorM():
    count = 0
    with open('DELHI.txt','r') as f:
    while True:
            line=f.readline()
            if not line:
                break
                    if   line[0]= = 'D' or line[0]= = 'M' :
                         count = count + 1
    if   count == 0 :
        print  ("no line starts with D or M")
    else:
        print  ("count=",count)
```

**28.** **(a)** (i) MIN (ISSUEDATE)

2016-12-13

(ii) DISTINCT DTYPE

Folk

Classical

Rock

(iii)

| DCODE | NAME | DTTTLE |
|---|---|---|
| R102 | AGAM SING | A day in life |
| F102 | ARTH JOSEPH | Universal soldier |
| FIOT | NISHA HANS | The planets |

(iv) DTITLE

A day in life

**(b)** SHOW CHARACTER SET

**29.**

```
def stringComp (str1,  str2):
    s1 = len(str1)
    s2 = len(str2)
    if  (s1 ! = s2) :
        return(False)
    else:
        for i in range s1
            if   str1 [i] ! = str2[i]:
                 return(False)
                 return  (True)
fstring=input ("Enter First string:")
```

```
sstring=input ("Enter Second string:")
if   stringComp (fstring, sstring):
     print ("Strings are same")
else:
     print ("Strings are different")
```

**30.**

```
def isempty(stk) :
     if   stk= = [ ]
          return True
     else:
          return False
def pushbook  (stk, item):
     stk.append(item)
     top=len (stk)-l
def displaybook(stk):
     if   isempty(stk):
          print("Stack Empty")
     else:
          top=len (stk)-1
          print ("book    no..... book name")
          for a in range (top,-l.-l):
               print  (stk[a])
stk=[ ]
top=None
while True:
     print ("book")
     print("1. add a book")
     print   ("2. display list")
     print   ("3. exit")
     ch=int   (input("enter your choice 1-3:") )
if   ch==1:
     bno=input("enter book no:")
     bname=input  ("enter book name")
     item=[bnp, bname]
     pushbook  (stk, item)
     input ( )
elif ch==2:
     displaybook(stk)
     input ( )
elif ch=3:
```

```
    break
else:
    print("invalid choice!")
    input()
```

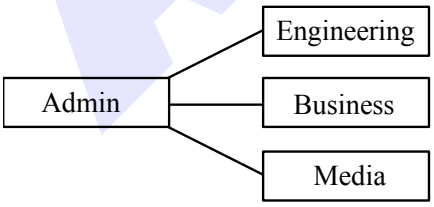
**OR**

```
def printlist (s) :
    i=0
    for i  in range (len(s) :
        print   (i,s[i] )
    i = 0
phonenumbers = ['9840012345', '9840011111', '9845622222', '9850012345', '9884412345']
    flag=0
number = raw_input("Enter the phone number to be searched")
number = number.strip( )
    try:
        i = phonenumbers.index(number)
        if  i >= 0 :
            flag=l
    except ValueError:
        pass
    if (flag < >0) :
        print ("\nphone number found in Phonebook at index", i)
    else:
            print ('\iphonenumbernotfoundin phonebook")
    print(  "\nPHONEBOOK")
printlist(phonenumbers)
```

## SECTION-D

**31.** (i) Admin (Due to maximum number of computers)

(ii)




(iii) Firewall or Router

(iv) Video Conferencing

(v) MAN (Metropolitan Area Network)

**32.** **(a)**

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
| 2 | 4 | 6 | 8 | 10 | 12 | 14 | 16 | 18 |
| 3 | 6 | 9 | 12 | 15 | 18 | 21 | 24 | 27 |
| 4 | 8 | 12 | 16 | 20 | 24 | 28 | 32 | 36 |
| 5 | 10 | 15 | 20 | 25 | 30 | 35 | 40 | 45 |
| 6 | 12 | 18 | 24 | 30 | 36 | 42 | 48 | 54 |
| 7 | 14 | 21 | 28 | 35 | 42 | 49 | 56 | 63 |
| 8 | 16 | 24 | 32 | 40 | 48 | 56 | 64 | 72 |
| 9 | 18 | 27 | 36 | 45 | 54 | 63 | 72 | 81 |

**(b)**

```
import MySQLdb
db = MySQLdb.connect(localhost','HRM an','HRmanapwd','compvtLtd')
cursor-db.cursor( )
cursor.execute("Drop Table IF Exists Employee")
sql="""Create Table Employee(*Emp_id'   INT NOTNULL, 'Emp_name' char(50)
NOTNULL,   'Dept' char(20)
NOTNULL,   'Age'  INT NOTNULL)"""
cursor.execute(sql)
cursor.execute("""create index eid on user,(Emp_Id)""")
cursor.commit( )
cursor.close( )
db. close()
```

**OR**

**(a)** **Output : 3**

**Working :**

x + = x – x will be calculated as x = 3 + 3 – 3

Hence, final value of x = 3

**(b)**

```
import MySQLdb
db=MySQLdb.connect("localhost", "Admin", "Ad123", "HMD")
cursor=db.cursor( )
cursor.execute("DROP TABLE IF EXISTS Location")
sql="""Create Table Location (id Numeric (5) PRIMARY KEY, bldg_code varchar(10)
Not Null, room varchar(6) Not Null, capacity, Numeric (5) Not Null)"""
cursor.execute(sql)
db. close()
```

**33.** Comma Separated Value

**PROGRAM**

```
def count_A_M( ):
    f = open("story.txt","r")
    A,M = 0, 0
    r = f.read()
    for x in r:
        if x[0]== "A" or x[0] == "a":
            A=A+1
        elif x[0] =="A" or x[0]=="m":
            M=M+1
    f.close( )
    print ("A or a:", A)
    print("M or m:", M)
```

**OR**

Need for a data file :

(i) It is a convenient way to deal with large quantities of data.

(ii) To avoid input of data multiple times during program execution.

**Program**

```
def  athletics ( ):
   file1 = open ("sports .dat", 'r')
   file2 = open ("Athletics.dat", 'w')
   rec = file1. readline( )
   while rec! = "":
      sport = rec.split ('n')
      if sport[0] == "Athletics":
         file2.write (rec)
         file2.write ('\n')
      else:
         pass
      rec = filel.readline( )
      file1.close()
      file2.close ()
   return
```

## SECTION-E

**34.** (i) ALTER TABLE STUDENTS DROP PHONE;

(ii) For the table Students primary key will be RNO and for the table sports primary key wll be admno.

(iii) (a) SELECT NAME, GAME FROM STUDENTS, SPORTS WHERE STUDENTS. ADMNO=SPORTS. ADMNO AND ADDRESS IS NOT NULL;

(b) SELECT NAME, COACHNAME FROM STUDENTS, SPORTS WHERE CLASS LIKE "12%" AND STUDENTS ADMNO = SPORTS. ADMNO;

**OR**

(iii) (a) SELECT COUNT(*) FROM STUDENTS, SPORTS WHERE GAME = "VOLLEYBALL" AND STUDENTS.ADMNO = SPORTS. ADMNO;

(b) SELECT NAME, COACHNAME FROM Students, sports where class like "12%" AND students. admno=sports.admno

**35.** (i) try (lmark)

(ii) load(f).

(iii) rec[0]==l. f.close()