Computer Science Journal

Jasveer Singh Dhillon
Board Roll No.: E223300310446
Class: XII B

Certificate

This is to certify that this computer journal has been completed by **Jasveer Singh Dhillon** of Class **XII-B**, in partial fulfilment of the curriculum of the CENTRAL BOARD OF SECONDARY EDUCATION leading to the award of the

SENIOR SECONDARY CERTIFICATE for AISSCE EXAMINATION for the year 2020-2021

_____ _____

External Examiner Internal Examiner

Date: Date:

_____ _____

SCHOOL SEAL PRINCIPAL

INDEX

1. Determine whether a number entered is a perfect number, an armstrong number or a palindrome.

```python
def palindrome(n):
    temp = n
    rev = 0
    while n > 0:
        dig = n % 10
        rev = rev * 10 + dig
        n = n // 10
    if temp == rev:
        print(temp, "The number is a palindrome!")
    else:
        print(temp, "The number isn't a palindrome!")

def armstrong(n):
    count = 0
    temp = n
    while temp > 0:
        digit = temp % 10
        count += digit ** 3
        temp //= 10
    if n == count:
        print(n, "is an Armstrong number")
    else:
        print(n, "is not an Armstrong number")

def perfect(n):
    count = 0
    for i in range(1, n):
        if n % i == 0:
            count = count + i
    if count == n:
        print(n, "The number is a Perfect number!")
    else:
        print(n, "The number is not a Perfect number!")

n = int(input("Enter number:"))
palindrome(n)
armstrong(n)
perfect(n)
```

```
Enter number:153
153 The number isn't a palindrome!
153 is an Armstrong number
153 The number is not a Perfect number!

Process finished with exit code 0
```

```
Enter number:99099
99099 The number is a palindrome!
99099 is not an Armstrong number
99099 The number is not a Perfect number!

Process finished with exit code 0
```

2. Input a list of numbers and swap elements at the even location with the elements at the odd location.

```python
mylist = []
print("Enter 5 elements for the list: ")
for i in range(5):
    value = int(input())
    mylist.append(value)
# printing original list
print("The original list : " + str(mylist))
# Separating odd and even index elements
odd_i = []
even_i = []
for i in range(0, len(mylist)):
    if i % 2:
        even_i.append(mylist[i])
    else :
        odd_i.append(mylist[i])
result = odd_i + even_i
# print result
print("Separated odd and even index list: " + str(result))
```

```
Enter 5 elements for the list:
534
5437
93287
43789
63498
The original list : [534, 5437, 93287, 43789, 63498]
Separated odd and even index list: [534, 93287, 63498, 5437, 43789]
```

3. Create a dictionary with the roll number, name and marks of n students in a class and display the names of students who have marks above 75.

```
no_of_std = int(input("Enter number of students: "))
result = {}
for i in range(no_of_std):
    print("Enter Details of student No.", i+1)
    roll_no = int(input("Roll No: "))
    std_name = input("Student Name: ")
    marks = int(input("Marks: "))
    result[roll_no] = [std_name, marks]

print(result)

# Display names of students who have got marks more than 75

for student in result:
    if result[student][1] > 75:
        print("Student's name who get more than 75 marks is/are",(result[student][0]))
```

```
Enter number of students: 4
Enter Details of student No. 1
Roll No: 1
Student Name: XYZ
Marks: 55
Enter Details of student No. 2
Roll No: 2
Student Name: ABC
Marks: 92
Enter Details of student No. 3
Roll No: 3
Student Name: PQR
Marks: 88
Enter Details of student No. 4
Roll No: 4
Student Name: MNO
Marks: 74
{1: ['XYZ', 55], 2: ['ABC', 92], 3: ['PQR', 88], 4: ['MNO', 74]}
Student's name who get more than 75 marks is/are ABC
Student's name who get more than 75 marks is/are PQR
```

4.Input a number and call a function primeornot() to check and display whether it is a prime number or not.

```python
def primeornot():
    n = int(input("Enter a number:"))
    if n > 1:

        for i in range(2, int(n / 2) + 1):

            if (n % i) == 0:
                print(n, "is not a prime number")
                break
        else:
            print(n, "is a prime number")
    else:
        print(n, "is not a prime number")


primeornot()
```

```
Enter a number:27644437
27644437 is a prime number

Process finished with exit code 0
```

5. Count and display the number of vowels, consonants, uppercase, lowercase characters in a user entered string.

```python
s = input("Enter any string :")
vowel = consonent = uppercase = lowercase = 0
for i in s:
    if (i == 'a' or i == 'e' or i == 'i' or i == 'o' or i == 'u' or i == 'A' or i == 'E' or i == 'I' or i == 'O' or i == 'U'):
        vowel = vowel + 1
    else:
        consonent = consonent + 1
    if i.isupper():
        uppercase = uppercase + 1

    if i.islower():
        lowercase = lowercase + 1

print("Total number of vowel:", vowel)
print("Total number of consonent:", consonent)
print("Total number of uppercase letter:", uppercase)
print("Total number of lowercase letter:", lowercase)
```

```
Enter any string :QWErtyUIOPpaSDFGhjkLZXCVBnm
Total number of vowel: 5
Total number of consonent: 22
Total number of uppercase letter: 17
Total number of lowercase letter: 10
```

6. Input a string and determine whether it is a palindrome or not without using slicing.

```python
def isPalindrome():
    s = input("Enter a string:")
    if s == s[::-1]:
        print('The string is a palindrome')
    else:
        print('The string is not a palindrome')


isPalindrome()
```

```
Enter a string:racecar
The string is a palindrome

Process finished with exit code 0
```

7. Write a random number generator that generates random numbers between 1 and 6

```
def random():
    import random
    s = random.randint(1, 6)
    return s


for i in range(6):
    print(random())
```

```
6
2
5
2
4
1

Process finished with exit code 0
```

8.Post writing, read the file and display
- the number of words in the file
- the number of special characters (including spaces) - the sum of the digits present in the file

Text Files Processing.py

```
prose = ''' \"Master, you are wonderful!\" A student, taking his leave, gazed
ardently at the patriarchal sage. \"You have renounced riches and
comforts to seek God and teach us wisdom!\" It was well-known that
Bhaduri Mahasaya had forsaken great family wealth in his early
childhood, when single-mindedly he entered the yogic path.122
\'You are reversing the case!\" The saint\'s face held a mild rebuke. \"I
have left a few paltry rupees, a few petty pleasures, for a cosmic
empire of endless bliss. How then have I denied myself anything? I
know the joy of sharing the treasure. Is that a sacrifice? The
shortsighted worldly folk are verily the real renunciates! They
relinquish an unparalleled divine possession for a poor handful of
earthly toys!\" '''


words = 0
digitSum = 0
spChar = 0

with open('Files.txt', 'w+') as f:
    f.write(prose)

    f.seek(0)
    for line in f.readlines():
        for word in line.split(' '):
            words += 1

    f.seek(0)
    for i in f.read():
        if i.isdigit():
            digitSum += int(i)
        elif i.isalpha():
            pass
        else:
            spChar += 1

    print('No. of words in Files.txt: ', words)
    print('No. of special characters in Files.txt: ', spChar)
    print('Sum of all digits in Files.txt: ', digitSum)
```

Output:

```
No. of words in Files.txt:  121
No. of special characters in Files.txt:  149
Sum of all digits in Files.txt:  5

Process finished with exit code 0
```

9.Write a program to remove all the words that contain the character `a' in a file and write it to another file.

Text Files Processing.py

```
l = []
finalword = ''
with open('Files.txt', 'r+') as f:
    data = f.read()
    for i in data.split():
        if 'a' in i:
            l.append(i)

with open('write.txt', 'w') as g:
    for word in l:
        for i in word:
            if i.isalpha():
                finalword += i
        g.write(finalword)
        g.write('\n')
        finalword = ''

```

Write.txt:

```
Master
are
taking
leave
gazed
ardently
at
patriarchal
sage
have
and
and
teach
was
that
Bhaduri
Mahasaya
had
forsaken
great
family
wealth
early
path
are
case
saints
face
a
have
a
paltry
a
pleasures
a
have
anything
sharing
treasure
that
a
sacrifice
are
real
renunciates
an
unparalleled
a
handful
earthly
```

10.

Binary Files Processing.py

```
import pickle


# Defining Functions
def recordAdd(name, cellno, email, file):
    record = [name, cellno, email]
    pickle.dump(record, file)


def recordSearch(name, file):
    found = False
    while not found:
        try:
            record = pickle.load(file)
            if record[0] == name:
                print('Name: ', record[0])
                print('Cell No.: ', record[1])
                print('Email ID: ', record[2])
                found = True
        except EOFError:
            print('No records found')
            break


# Taking Inputs
file = input('File name: ')
print('What do you want to do:')
print('1] Add Records to file')
print('2] Search Record from file')
x = int(input('Select an option: '))
```

```
b = 0

if x == 1:
    with open(file, 'ab') as f:
        while True:
            b = input('Cell No. (enter an alphabet to end): ')
            if not b.isdigit():
                break
            a = input('Name : ')
            c = input('Email ID:')
            recordAdd(a, b, c, f)
elif x == 2:
    with open(file, 'rb') as f:
        a = input('Search Name: ')
        recordSearch(a, f)
else:
    print('Invalid Option')
    x = int(input('Please select a valid option: '))

```

Output:

Adding Records:

```
File name: binary.dat
What do you want to do:
1] Add Records to file
2] Search Record from file
Select an option: 1
Cell No. (enter an alphabet to end): 11
Name : Prisoner Sheesh
Email ID:sheesh@email.com
Cell No. (enter an alphabet to end): 27
Name : CheeseBit27
Email ID:cheezymail@email.com
Cell No. (enter an alphabet to end): f

Process finished with exit code 0
```

Searching Records:

```
File name: binary.dat
What do you want to do:
1] Add Records to file
2] Search Record from file
Select an option: 2
Search Name: Prisoner Sheesh
Name:  Prisoner Sheesh
Cell No.:  11
Email ID:  sheesh@email.com

Process finished with exit code 0
```

11.

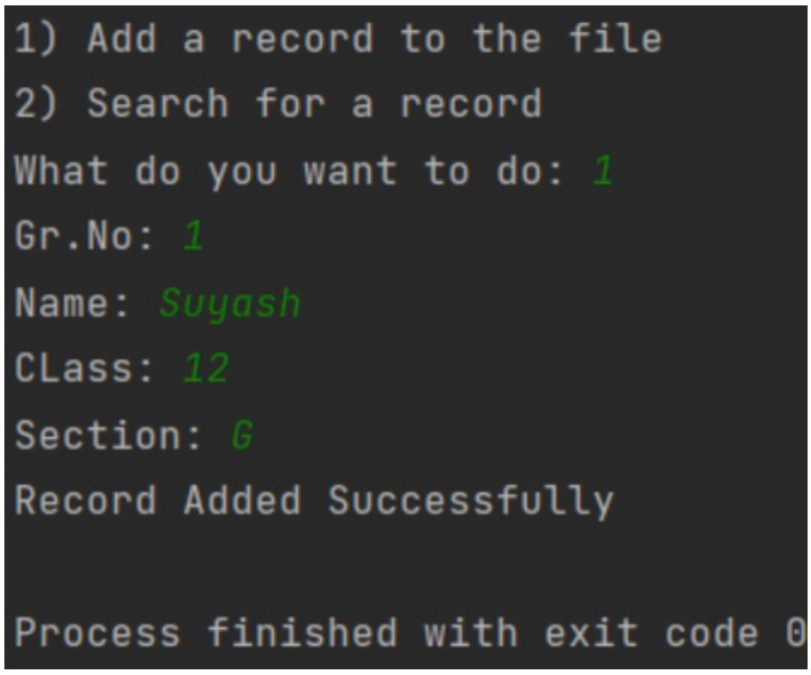
12.

main.py

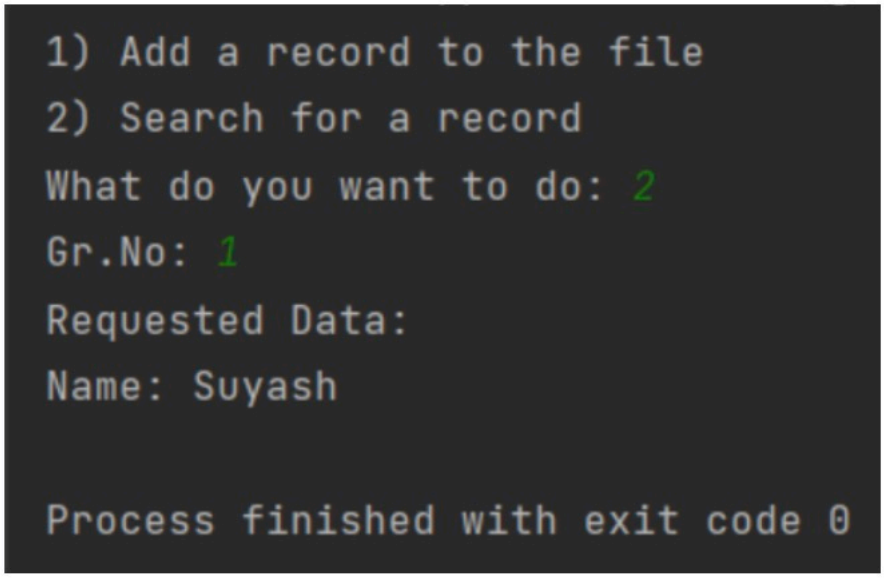
```
import csv


# Functions
def add_record(grno, name, clas5, section):
    l = []
    f = open('csv.csv', 'w+')
    fw = csv.writer(f)
    fr = csv.reader(f)

    for a in fr:
        l.append(a)
    a = [grno, name, clas5, section]
    l.append(a)

    for i in l:
        fw.writerow(i)
    print('Record Added Successfully')


def read_record(grno):
    found = 0
    f = open('csv.csv', 'r')
    fr = csv.reader(f)

    for a in fr:
        try:
            print('trying', a)
            if a[0] == str(grno):
                print('Requested Data:')
                print('Name:', a[1])
                found += 1
        except IndexError:
            if not found:
                print('No Records Found')

# Menu
print('1) Add a record to the file')
print('2) Search for a record')
x = int(input('What do you want to do: '))

if x == 1:
    grno = int(input('Gr.No: '))
    name = input('Name: ')
    clas5 = int(input('CLass: '))
    section = input('Section: ')
    add_record(grno, name, clas5, section)
elif x == 2:
    grno = int(input('Gr.No: '))
    read_record(grno)
```

Output:

Adding a record:



Searching a record:



13.

Stack using List.py

```
class Stack:
    def __init__(self):
        self.list = ['person1', 2, [1, 'hangman']]

    def push(self, value):
        self.list.append(value)
        print(f'{value} has been pushed into the stack.\nThe new stack is {self.list}')

    def pop(self):
        if self.list:
            a = self.list.pop()
            print(f'{a} has been removed from the stack.\nThe new stack is {self.list}')
        else:
            print('the Stack is empty')

    def search(self, value):
        for i in self.list:
            if str(i) == value:
                print(f'{value} is present at position {self.list.index(i)}')
                break
        else:
            print(f'{value} is not present in the stack')

    def display(self):
        print('Stack:', self.list)


stack = Stack()
print('1) Insert a Value \n2) Remove a Value \n3) Display a Value\n4) Search a Value')
x = int(input('What do you want to do:'))
```

```

if x == 1:
    value = input('Value to be inserted: ')
    stack.push(value)
elif x == 2:
    stack.pop()
elif x == 3:
    stack.display()
elif x == 4:
    value = input('Value: ')
    stack.search(value)
else:
    print('Requested service not available')

```

Output:

```
1) Insert a Value
2) Remove a Value
3) Display a Value
4) Search a Value
What do you want to do:1
Value to be inserted: new value
new value has been pushed into the stack.
The new stack is ['person1', 2, [1, 'hangman'], 'new value']

Process finished with exit code 0
```

```
1) Insert a Value
2) Remove a Value
3) Display a Value
4) Search a Value
What do you want to do:2
[1, 'hangman'] has been removed from the stack.
The new stack is ['person1', 2]

Process finished with exit code 0
```

```
1) Insert a Value
2) Remove a Value
3) Display a Value
4) Search a Value
What do you want to do:3
Stack: ['person1', 2, [1, 'hangman']]

Process finished with exit code 0
```

```
1) Insert a Value
2) Remove a Value
3) Display a Value
4) Search a Value
What do you want to do:4
Value: 2
2 is present at position 1
```

14.

Queue using List.py

```
class Queue:
    def __init__(self):
        self.list = ['hello', 34, [1, 'hi']]

    def enqueue(self, value):
        self.list.append(value)
        print(f'{value} has been enqueued to the queue.\nThe new queue is {self.list}')

    def dequeue(self):
        if self.list:
            print(f'{self.list[0]} has been dequeued from the queue.')
            del self.list[0]
            print(f'The new queue is {self.list}')
        else:
            print('The Queue is empty')

    def display(self):
        print(self.list)

    def search(self, value):
        for i in self.list:
            if str(i) == value:
                print(f'{value} is in the queue at the position {self.list.index(i) + 1}')
                break
        else:
            print(f'{value} is not in the queue')


queue = Queue()
print('1) Add a Value\n2)Remove a Value\n3)Display the Queue\n4)Search a Value')
```

```
x = int(input('What do you want to do:'))

if x == 1:
    value = input('Value to be inserted:')
    queue.enqueue(value)
elif x == 2:
    queue.dequeue()
elif x == 3:
    queue.display()
elif x == 4:
    value = input('Value to be searched:')
    queue.search(value)
else:
    print('Requested function not available')
```

Output:

Task1: Enqueue

```
1) Add a Value
2)Remove a Value
3)Display the Queue
4)Search a Value
What do you want to do:1
Value to be inserted:value
value has been enqueued to the queue.
The new queue is ['hello', 34, [1, 'hi'], 'value']
```

Task2: Dequeue

```
1) Add a Value
2)Remove a Value
3)Display the Queue
4)Search a Value
What do you want to do:2
hello has been dequeued from the queue.
The new queue is [34, [1, 'hi']]

Process finished with exit code 0
```

Task3: Display the queue

```
1) Add a Value
2)Remove a Value
3)Display the Queue
4)Search a Value
What do you want to do:3
['hello', 34, [1, 'hi']]

Process finished with exit code 0
```

Task4: Search for a value in the queue

```
1) Add a Value
2)Remove a Value
3)Display the Queue
4)Search a Value
What do you want to do:4
Value to be searched:[1, 'hi']
[1, 'hi'] is in the queue at the position 3
```

15.

MySQL Connector 1.py

```
import mysql.connector
mydb = mysql.connector.connect(host='localhost', user='root', password='Cheesebit', database='school')
mycursor = mydb.cursor()

x = input('Name of the Student:')
topics = ['No.', 'Name', 'Stipend', 'Stream', 'AvgMark', 'Grade', 'Class']
j = 0

mycursor.execute(f'select * from student where Name = "{x}"')
for x in mycursor:
    for i in x:
        print(topics[j], ' : ', i)
        j += 1
```

Run: MySQL Connector 1 (1)

```
Users/jasveerdhillon/Python/PyCharm/XII Computer Science Journal/bin/python"
Name of the Student:Karan
No.  :  1.
Name  :  Karan
Stipend  :  400.0
Stream  :  Medical
AvgMark  :  78.5
Grade  :  B
Class  :  12B

Process finished with exit code 0
```

16.

```
import mysql.connector
mydb = mysql.connector.connect(host='localhost', user='root', password='Cheesebit', database='school')
mycursor = mydb.cursor()

a = int(input('No.:'))
b = input('Name of the Student:')
c = float(input('Stipend:'))
d = input('Stream:')
e = float(input('AvgMark:'))
f = input('Grade:')
g = input('Class:')
a = str(a) + '.'
mycursor.execute(f'insert into student values ("{a}", "{b}", {c}, "{d}", {e}, "{f}", "{g}")')
mydb.commit()
```

```
mysql> ;select * from student;
ERROR:
No query specified

+-----+--------+---------+---------+---------+-------+-------+
| No  | Name   | Stipend | Stream  | AvgMark | Grade | Class |
+-----+--------+---------+---------+---------+-------+-------+
| 1.  | Karan  |  400.00 | Medical |    78.5 | B     | 12B   |
| 2.  | Mohit  |  450.00 | Comm.   |    89.2 | A     | 11C   |
| 3.  | Divya  |  300.00 | Comm.   |    68.6 | C     | 12C   |
| 4.  | Arun   |  350.00 | Arts    |    73.1 | B     | 12D   |
| 5.  | John   |  500.00 | NMed    |    90.6 | A     | 11A   |
| 6.  | Mohan  |  400.00 | Medical |    75.4 | B     | 12B   |
| 7.  | Vikas  |  250.00 | Arts    |    64.4 | C     | 11A   |
| 8.  | Akash  |  450.00 | NMed    |    88.5 | A     | 12A   |
| 9.  | Bobby  |  500.00 | NMed    |    92.0 | A     | 12A   |
| 10. | Abha   |  300.00 | Comm.   |    67.5 | C     | 12C   |
+-----+--------+---------+---------+---------+-------+-------+
10 rows in set (0.00 sec)

mysql>
```

Output:

```
/Users/ashishnanda/Desktop/Project-Arul/bin/python "/Users/ashishnanda/Library/Application Support/JetBra
 KMR/School/Computer Practical File Contents/MySQL Connector 1.py"
No.:11
Name of the Student:Sysh
Stipend:450
Stream:CS
AvgMark:99
Grade:A
Class:12G

Process finished with exit code 0
```

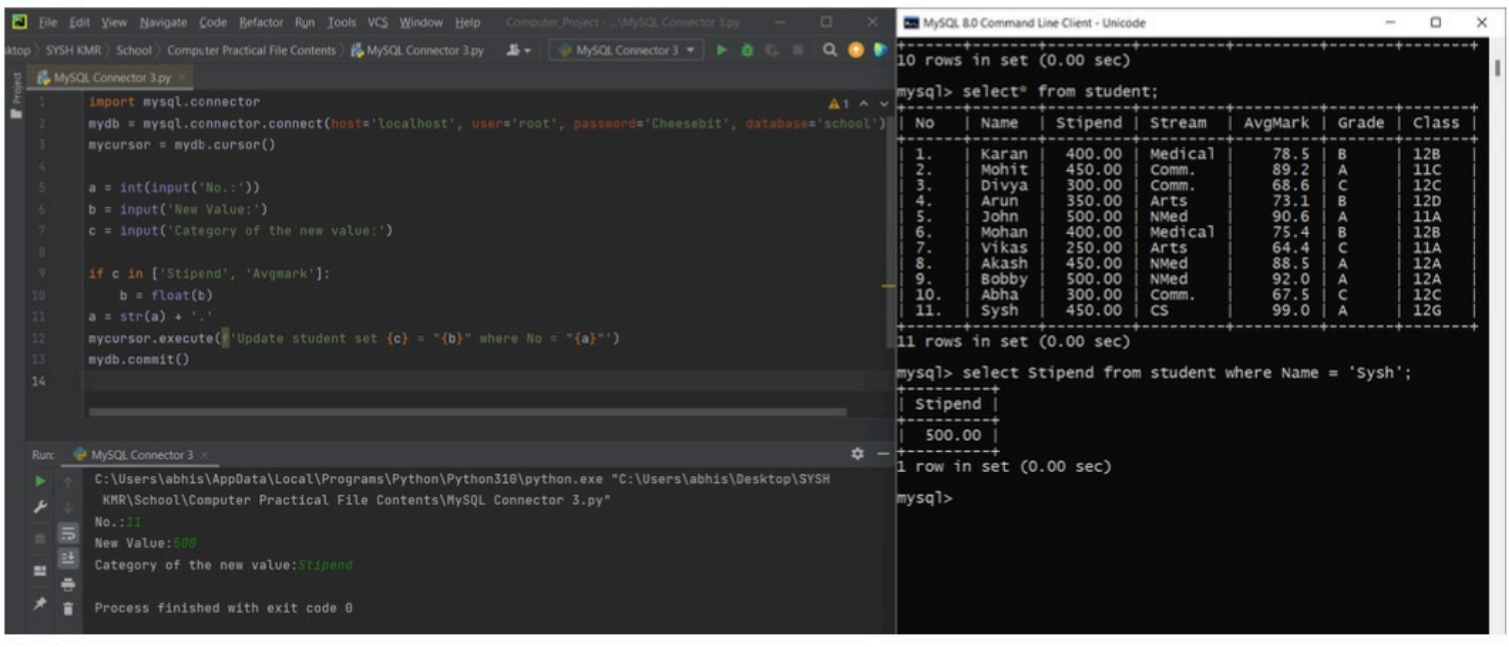
```
mysql> ;select * from student;
ERROR:
No query specified

+-----+--------+---------+---------+---------+-------+-------+
| No  | Name   | Stipend | Stream  | AvgMark | Grade | Class |
+-----+--------+---------+---------+---------+-------+-------+
| 1.  | Karan  |  400.00 | Medical |    78.5 | B     | 12B   |
| 2.  | Mohit  |  450.00 | Comm.   |    89.2 | A     | 11C   |
| 3.  | Divya  |  300.00 | Comm.   |    68.6 | C     | 12C   |
| 4.  | Arun   |  350.00 | Arts    |    73.1 | B     | 12D   |
| 5.  | John   |  500.00 | NMed    |    90.6 | A     | 11A   |
| 6.  | Mohan  |  400.00 | Medical |    75.4 | B     | 12B   |
| 7.  | Vikas  |  250.00 | Arts    |    64.4 | C     | 11A   |
| 8.  | Akash  |  450.00 | NMed    |    88.5 | A     | 12A   |
| 9.  | Bobby  |  500.00 | NMed    |    92.0 | A     | 12A   |
| 10. | Abha   |  300.00 | Comm.   |    67.5 | C     | 12C   |
+-----+--------+---------+---------+---------+-------+-------+
10 rows in set (0.00 sec)

mysql> select* from student;
+-----+--------+---------+---------+---------+-------+-------+
| No  | Name   | Stipend | Stream  | AvgMark | Grade | Class |
+-----+--------+---------+---------+---------+-------+-------+
| 1.  | Karan  |  400.00 | Medical |    78.5 | B     | 12B   |
| 2.  | Mohit  |  450.00 | Comm.   |    89.2 | A     | 11C   |
| 3.  | Divya  |  300.00 | Comm.   |    68.6 | C     | 12C   |
| 4.  | Arun   |  350.00 | Arts    |    73.1 | B     | 12D   |
| 5.  | John   |  500.00 | NMed    |    90.6 | A     | 11A   |
| 6.  | Mohan  |  400.00 | Medical |    75.4 | B     | 12B   |
| 7.  | Vikas  |  250.00 | Arts    |    64.4 | C     | 11A   |
| 8.  | Akash  |  450.00 | NMed    |    88.5 | A     | 12A   |
| 9.  | Bobby  |  500.00 | NMed    |    92.0 | A     | 12A   |
| 10. | Abha   |  300.00 | Comm.   |    67.5 | C     | 12C   |
| 11. | Sysh   |  450.00 | CS      |    99.0 | A     | 12G   |
+-----+--------+---------+---------+---------+-------+-------+
11 rows in set (0.00 sec)

mysql>
```

17.

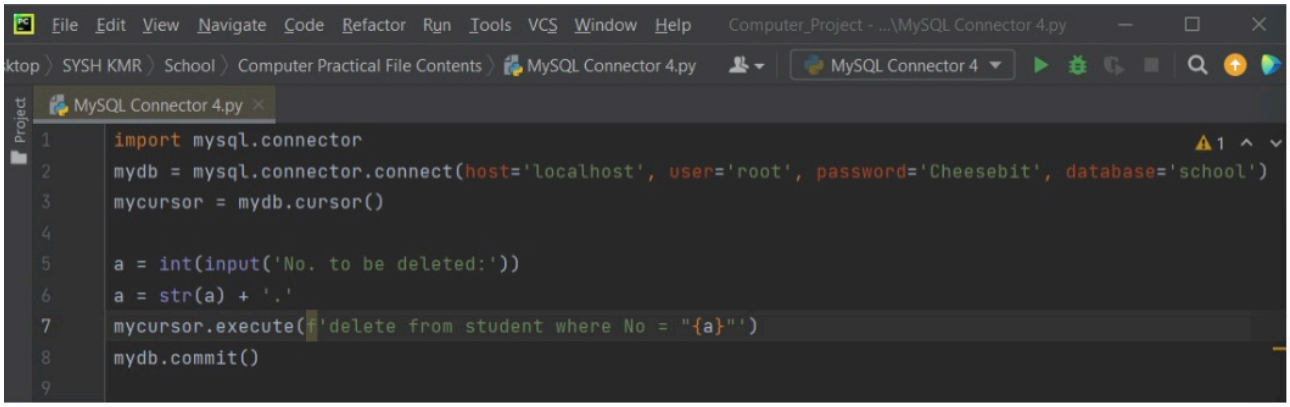
```python
import mysql.connector
mydb = mysql.connector.connect(host='localhost', user='root', password='Cheesebit', database='school')
mycursor = mydb.cursor()

a = int(input('No.:'))
b = input('New Value:')
c = input('Category of the new value:')

if c in ['Stipend', 'Avgmark']:
    b = float(b)
a = str(a) + '.'
mycursor.execute(f'Update student set {c} = "{b}" where No = "{a}"')
mydb.commit()
```

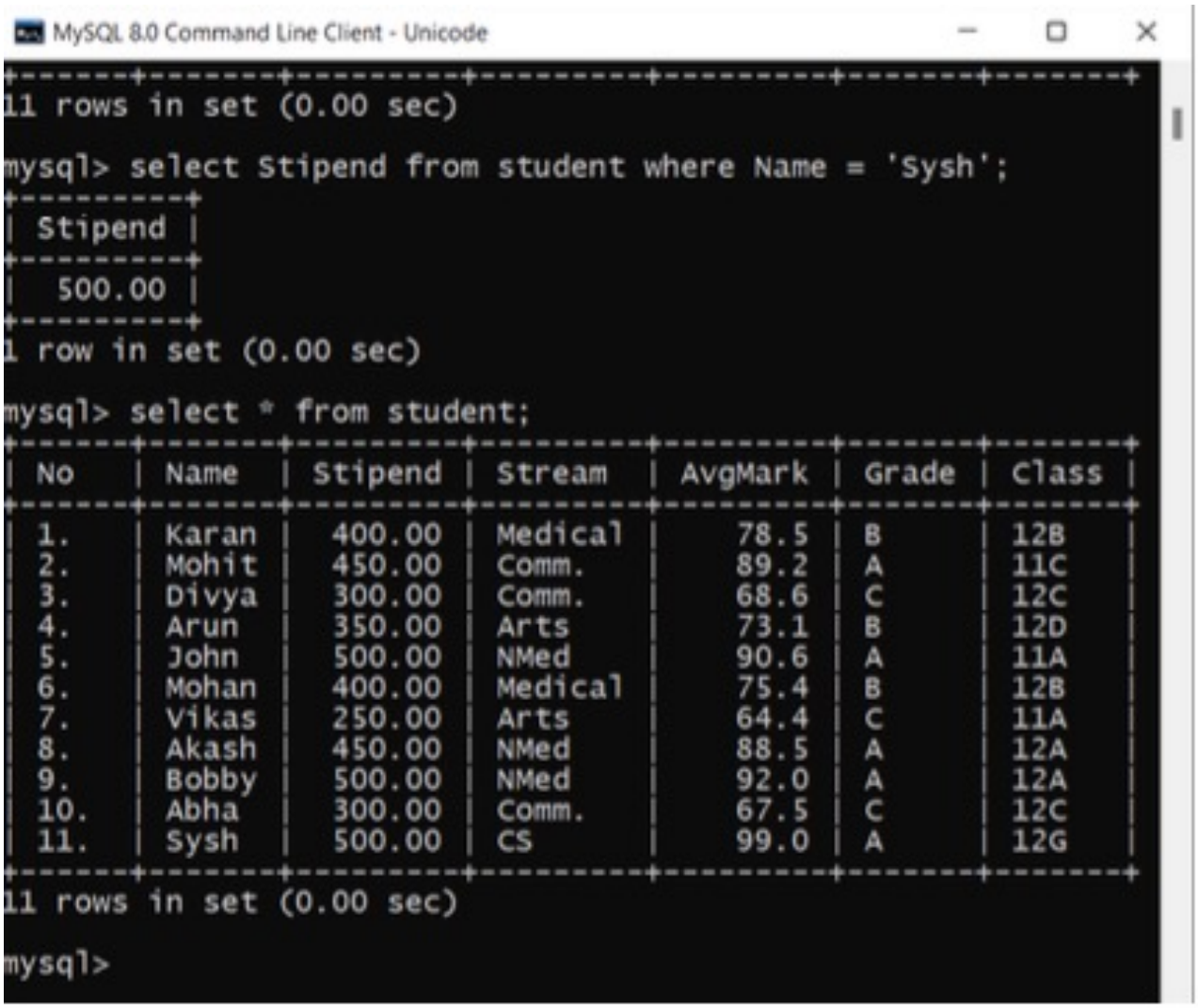
```
mysql> select* from student;
+------+--------+---------+---------+---------+-------+-------+
| No   | Name   | Stipend | Stream  | AvgMark | Grade | Class |
+------+--------+---------+---------+---------+-------+-------+
| 1.   | Karan  |  400.00 | Medical |    78.5 | B     | 12B   |
| 2.   | Mohit  |  450.00 | Comm.   |    89.2 | A     | 11C   |
| 3.   | Divya  |  300.00 | Comm.   |    68.6 | C     | 12C   |
| 4.   | Arun   |  350.00 | Arts    |    73.1 | B     | 12D   |
| 5.   | John   |  500.00 | NMed    |    90.6 | A     | 11A   |
| 6.   | Mohan  |  400.00 | Medical |    75.4 | B     | 12B   |
| 7.   | Vikas  |  250.00 | Arts    |    64.4 | C     | 11A   |
| 8.   | Akash  |  450.00 | NMed    |    88.5 | A     | 12A   |
| 9.   | Bobby  |  500.00 | NMed    |    92.0 | A     | 12A   |
| 10.  | Abha   |  300.00 | Comm.   |    67.5 | C     | 12C   |
| 11.  | Sysh   |  450.00 | CS      |    99.0 | A     | 12G   |
+------+--------+---------+---------+---------+-------+-------+
11 rows in set (0.00 sec)

mysql>
```

Output:

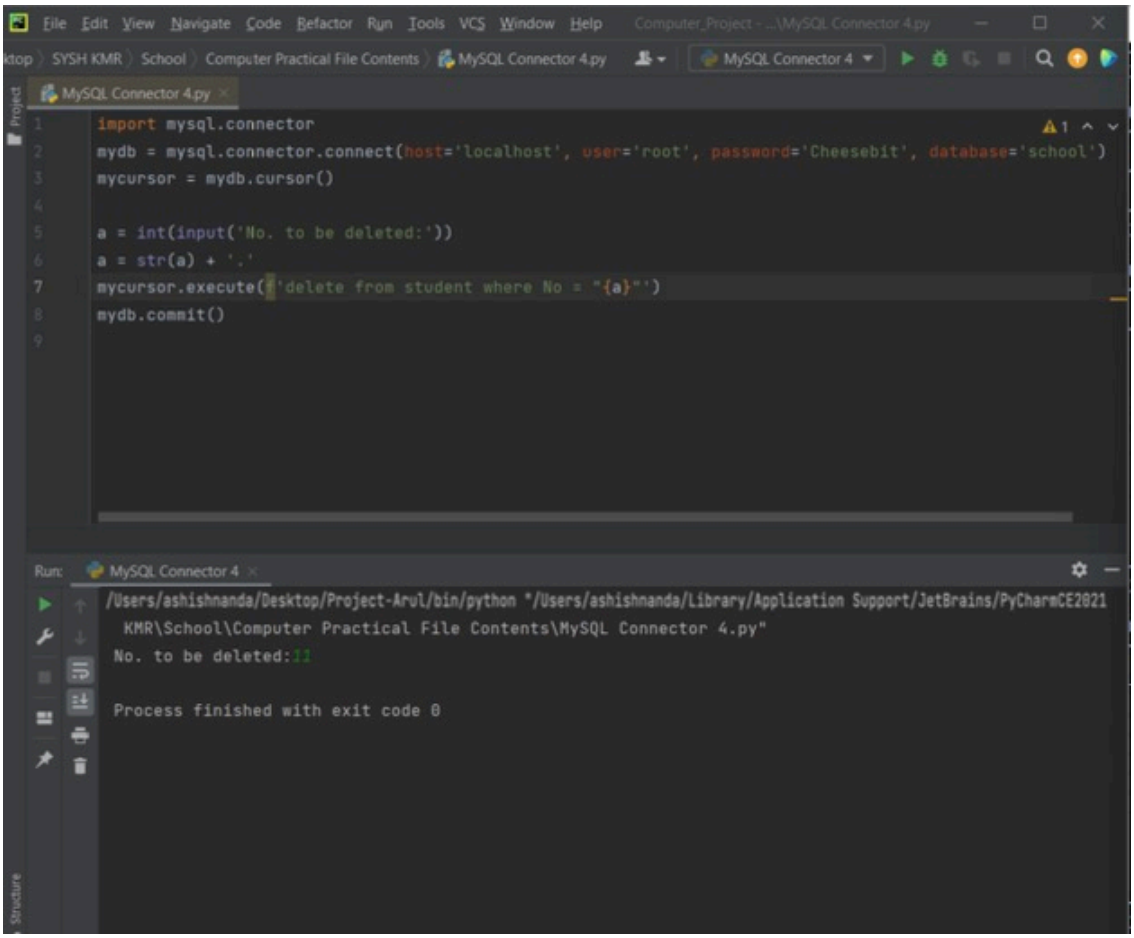


18.


File Edit View Navigate Code Refactor Run Tools VCS Window Help
Computer_Project - ...\MySQL Connector 4.py
SYSH KMR School Computer Practical File Contents MySQL Connector 4.py
MySQL Connector 4
MySQL Connector 4.py
import mysql.connector
mydb = mysql.connector.connect(host='localhost', user='root', password='Cheesebit', database='school')
mycursor = mydb.cursor()
a = int(input('No. to be deleted:'))
a = str(a) + '.'
mycursor.execute(f'delete from student where No = "{a}"')
mydb.commit()


MySQL 8.0 Command Line Client - Unicode
11 rows in set (0.00 sec)
mysql> select Stipend from student where Name = 'Sysh';
| Stipend |
| 500.00 |
1 row in set (0.00 sec)
mysql> select * from student;
No	Name	Stipend	Stream	AvgMark	Grade	Class
1.	Karan	400.00	Medical	78.5	B	12B
2.	Mohit	450.00	Comm.	89.2	A	11C
3.	Divya	300.00	Comm.	68.6	C	12C
4.	Arun	350.00	Arts	73.1	B	12D
5.	John	500.00	NMed	90.6	A	11A
6.	Mohan	400.00	Medical	75.4	B	12B
7.	Vikas	250.00	Arts	64.4	C	11A
8.	Akash	450.00	NMed	88.5	A	12A
9.	Bobby	500.00	NMed	92.0	A	12A
10.	Abha	300.00	Comm.	67.5	C	12C
11.	Sysh	500.00	CS	99.0	A	12G
11 rows in set (0.00 sec)
mysql>

Output:



```
import mysql.connector
mydb = mysql.connector.connect(host='localhost', user='root', password='Cheesebit', database='school')
mycursor = mydb.cursor()

a = int(input('No. to be deleted:'))
a = str(a) + '.'
mycursor.execute(f'delete from student where No = "{a}"')
mydb.commit()
```

```
/Users/ashishnanda/Desktop/Project-Arul/bin/python "/Users/ashishnanda/Library/Application Support/JetBrains/PyCharmCE2021
  KMR\School\Computer Practical File Contents\MySQL Connector 4.py"
No. to be deleted:11

Process finished with exit code 0
```

MySQL 8.0 Command Line Client - Unicode

```
11 rows in set (0.00 sec)

mysql> select Stipend from student where Name = 'Sysh';
+---------+
| Stipend |
+---------+
|  500.00 |
+---------+
1 row in set (0.00 sec)

mysql> select * from student;
+------+-------+---------+---------+---------+-------+-------+
| No   | Name  | Stipend | Stream  | AvgMark | Grade | Class |
+------+-------+---------+---------+---------+-------+-------+
| 1.   | Karan |  400.00 | Medical |    78.5 | B     | 12B   |
| 2.   | Mohit |  450.00 | Comm.   |    89.2 | A     | 11C   |
| 3.   | Divya |  300.00 | Comm.   |    68.6 | C     | 12C   |
| 4.   | Arun  |  350.00 | Arts    |    73.1 | B     | 12D   |
| 5.   | John  |  500.00 | NMed    |    90.6 | A     | 11A   |
| 6.   | Mohan |  400.00 | Medical |    75.4 | B     | 12B   |
| 7.   | Vikas |  250.00 | Arts    |    64.4 | C     | 11A   |
| 8.   | Akash |  450.00 | NMed    |    88.5 | A     | 12A   |
| 9.   | Bobby |  500.00 | NMed    |    92.0 | A     | 12A   |
| 10.  | Abha  |  300.00 | Comm.   |    67.5 | C     | 12C   |
| 11.  | Sysh  |  500.00 | CS      |    99.0 | A     | 12G   |
+------+-------+---------+---------+---------+-------+-------+
11 rows in set (0.00 sec)

mysql> select* from student;
+------+-------+---------+---------+---------+-------+-------+
| No   | Name  | Stipend | Stream  | AvgMark | Grade | Class |
+------+-------+---------+---------+---------+-------+-------+
| 1.   | Karan |  400.00 | Medical |    78.5 | B     | 12B   |
| 2.   | Mohit |  450.00 | Comm.   |    89.2 | A     | 11C   |
| 3.   | Divya |  300.00 | Comm.   |    68.6 | C     | 12C   |
| 4.   | Arun  |  350.00 | Arts    |    73.1 | B     | 12D   |
| 5.   | John  |  500.00 | NMed    |    90.6 | A     | 11A   |
| 6.   | Mohan |  400.00 | Medical |    75.4 | B     | 12B   |
| 7.   | Vikas |  250.00 | Arts    |    64.4 | C     | 11A   |
| 8.   | Akash |  450.00 | NMed    |    88.5 | A     | 12A   |
| 9.   | Bobby |  500.00 | NMed    |    92.0 | A     | 12A   |
| 10.  | Abha  |  300.00 | Comm.   |    67.5 | C     | 12C   |
+------+-------+---------+---------+---------+-------+-------+
10 rows in set (0.00 sec)
```

19. SQL I

1. Create the tables with the fields as shown above.
2. Insert the data into the table as per the table given.

```
mysql> create table STOCK(Item_no smallint, Item varchar(17), DCode smallint, Qty smallint, UnitPrice smallint, StockDate date);
Query OK, 0 rows affected (0.42 sec)

mysql> insert into stock values(5005, 'Ball pen 0.5', 102, 100, 16, '10-3-31');
Query OK, 1 row affected (0.09 sec)
```

```
mysql> create table DEALERS(DCode smallint, DName varchar(19));
Query OK, 0 rows affected (0.35 sec)

mysql> insert into DEALERS values(101, 'Reliable Stationars');
Query OK, 1 row affected (0.09 sec)
```

3. To display details of all items in the Stock table in ascending order of item.

```
mysql> select * from STOCK order by Item;
+---------+-------------------+-------+------+-----------+------------+
| Item_no | Item              | DCode | Qty  | UnitPrice | StockDate  |
+---------+-------------------+-------+------+-----------+------------+
|    5003 | Ball pen 0.25     |   102 |  150 |        20 | 2010-01-01 |
|    5005 | Ball pen 0.5      |   102 |  100 |        16 | 2010-03-31 |
|    5004 | Eraser Big        |   102 |   60 |        10 | 0009-12-12 |
|    5001 | Eraser Small      |   102 |  210 |         5 | 0009-03-19 |
|    5006 | Gel Pen Classic   |   101 |  200 |        22 | 0009-01-01 |
|    5002 | Gel Pen Premium   |   101 |  125 |        14 | 2010-02-14 |
|    5009 | Sharpener Classic |   103 |  160 |         8 | 0009-01-23 |
+---------+-------------------+-------+------+-----------+------------+
7 rows in set (0.06 sec)
```

4. To display ItemNo and Itemname of those items from stock table whose unit price is more than Rupees 10.

```
mysql> select Item_no, Item ItemName from STOCK where UnitPrice > 10;
+---------+-----------------+
| Item_no | ItemName        |
+---------+-----------------+
|    5005 | Ball pen 0.5    |
|    5003 | Ball pen 0.25   |
|    5002 | Gel Pen Premium |
|    5006 | Gel Pen Classic |
+---------+-----------------+
4 rows in set (0.00 sec)
```

5. To display the details of those items whose dealer code (Dcode) is 102 or Quantity in stock (Qty) is more than 100 from the table stock.

```
mysql> select * from STOCK where DCode = 102 or Qty > 100;
+---------+-------------------+-------+------+-----------+------------+
| Item_no | Item              | DCode | Qty  | UnitPrice | StockDate  |
+---------+-------------------+-------+------+-----------+------------+
|    5005 | Ball pen 0.5      |   102 |  100 |        16 | 2010-03-31 |
|    5003 | Ball pen 0.25     |   102 |  150 |        20 | 2010-01-01 |
|    5002 | Gel Pen Premium   |   101 |  125 |        14 | 2010-02-14 |
|    5006 | Gel Pen Classic   |   101 |  200 |        22 | 0009-01-01 |
|    5001 | Eraser Small      |   102 |  210 |         5 | 0009-03-19 |
|    5004 | Eraser Big        |   102 |   60 |        10 | 0009-12-12 |
|    5009 | Sharpener Classic |   103 |  160 |         8 | 0009-01-23 |
+---------+-------------------+-------+------+-----------+------------+
7 rows in set (0.00 sec)
```

6. To display Max. Unit price of items for each dealer individually as per Dcode from the table Stock.

```
mysql> select DCode ,max(UnitPrice) MRP from STOCK group by DCode;
+-------+------+
| DCode | MRP  |
+-------+------+
|   102 |   20 |
|   101 |   22 |
|   103 |    8 |
+-------+------+
3 rows in set (0.00 sec)
```

7. Increase Qty by 50 for all items.

```
mysql> Update STOCK set Qty = Qty + 50 where DCode in (101, 102, 103);
Query OK, 7 rows affected (0.07 sec)
Rows matched: 7  Changed: 7  Warnings: 0

mysql> select * from STOCK;
+---------+-------------------+-------+------+-----------+------------+
| Item_no | Item              | DCode | Qty  | UnitPrice | StockDate  |
+---------+-------------------+-------+------+-----------+------------+
|    5005 | Ball pen 0.5      |   102 |  150 |        16 | 2010-03-31 |
|    5003 | Ball pen 0.25     |   102 |  200 |        20 | 2010-01-01 |
|    5002 | Gel Pen Premium   |   101 |  175 |        14 | 2010-02-14 |
|    5006 | Gel Pen Classic   |   101 |  250 |        22 | 0009-01-01 |
|    5001 | Eraser Small      |   102 |  260 |         5 | 0009-03-19 |
|    5004 | Eraser Big        |   102 |  110 |        10 | 0009-12-12 |
|    5009 | Sharpener Classic |   103 |  210 |         8 | 0009-01-23 |
+---------+-------------------+-------+------+-----------+------------+
7 rows in set (0.00 sec)
```

8. Display the various items stored in the table.

```
mysql> select Item from STOCK;
+--------------------+
| Item               |
+--------------------+
| Ball pen 0.5       |
| Ball pen 0.25      |
| Gel Pen Premium    |
| Gel Pen Classic    |
| Eraser Small       |
| Eraser Big         |
| Sharpener Classic  |
+--------------------+
7 rows in set (0.00 sec)
```

9.Display the average qty for the items in the table.

```
mysql> select DCode ,avg(Qty) Average_Quantity from STOCK group by DCode;
+-------+------------------+
| DCode | Average_Quantity |
+-------+------------------+
|   102 |         180.0000 |
|   101 |         212.5000 |
|   103 |         210.0000 |
+-------+------------------+
3 rows in set (0.00 sec)
```

10.Count the number of items for each DCode.

```
mysql> select DCode, count(Item) No_of_Items from STOCK group by DCode;
+-------+-------------+
| DCode | No_of_Items |
+-------+-------------+
|   102 |           4 |
|   101 |           2 |
|   103 |           1 |
+-------+-------------+
3 rows in set (0.00 sec)
```

SQL II

1. Create table STUDENT with the fields as shown.

```
mysql> create table Student (No varchar(3), Name varchar(5), Stipend float(5,2), Stream varchar(7), AvgMark float(3,1), Grade char(1), Class char(3));
Query OK, 0 rows affected, 2 warnings (0.28 sec)

mysql> insert into Student values('1.', 'Karan', 400, 'Medical', 78.5, 'B', '12B');
Query OK, 1 row affected (0.06 sec)
```

2. Insert the data into the table as per the table given.

No	Name	Stipend	Stream	AvgMark	Grade	Class
1.	Karan	400.00	Medical	78.5	B	12B
2.	Mohit	450.00	Comm.	89.2	A	11C
3.	Divya	300.00	Comm.	68.6	C	12C
4.	Arun	350.00	Arts	73.1	B	12D
5.	John	500.00	NMed	90.6	A	11A
6.	Mohan	400.00	Medical	75.4	B	12B
7.	Vikas	250.00	Arts	64.4	C	11A
8.	Akash	450.00	NMed	88.5	A	12A
9.	Bobby	500.00	NMed	92.0	A	12A
10.	Abha	300.00	Comm.	67.5	C	12C

3. Select all the stream names from STUDENT.

```
mysql> select distinct(Stream) from Student;
+---------+
| Stream  |
+---------+
| Medical |
| Comm.   |
| Arts    |
| NMed    |
+---------+
4 rows in set (0.00 sec)
```

4. List the names of those students who are in class 12 sorted by stipend.

```
mysql> select Name from Student where Class like '12_' order by Stipend;
+-------+
| Name  |
+-------+
| Divya |
| Abha  |
| Arun  |
| Karan |
| Mohan |
| Akash |
| Bobby |
+-------+
7 rows in set (0.01 sec)
```

5. List all students sorted by AvgMark in descending order.

```
mysql> select * from Student order by AvgMark desc;
+------+-------+---------+---------+---------+-------+-------+
| No   | Name  | Stipend | Stream  | AvgMark | Grade | Class |
+------+-------+---------+---------+---------+-------+-------+
| 9.   | Bobby |  500.00 | NMed    |    92.0 | A     | 12A   |
| 5.   | John  |  500.00 | NMed    |    90.6 | A     | 11A   |
| 2.   | Mohit |  450.00 | Comm.   |    89.2 | A     | 11C   |
| 8.   | Akash |  450.00 | NMed    |    88.5 | A     | 12A   |
| 1.   | Karan |  400.00 | Medical |    78.5 | B     | 12B   |
| 6.   | Mohan |  400.00 | Medical |    75.4 | B     | 12B   |
| 4.   | Arun  |  350.00 | Arts    |    73.1 | B     | 12D   |
| 3.   | Divya |  300.00 | Comm.   |    68.6 | C     | 12C   |
| 10.  | Abha  |  300.00 | Comm.   |    67.5 | C     | 12C   |
| 7.   | Vikas |  250.00 | Arts    |    64.4 | C     | 11A   |
+------+-------+---------+---------+---------+-------+-------+
10 rows in set (0.00 sec)
```

6. Display a report listing name, stipend, stream and amount of stipend received in a year assuming that the stipend is paid every month.

```
mysql> select Name, Stipend, Stream, Stipend*12 Stipend_Recieved from Student;
+-------+---------+---------+-----------------+
| Name  | Stipend | Stream  | Stipend_Recieved |
+-------+---------+---------+-----------------+
| Karan |  400.00 | Medical |         4800.00 |
| Mohit |  450.00 | Comm.   |         5400.00 |
| Divya |  300.00 | Comm.   |         3600.00 |
| Arun  |  350.00 | Arts    |         4200.00 |
| John  |  500.00 | NMed    |         6000.00 |
| Mohan |  400.00 | Medical |         4800.00 |
| Vikas |  250.00 | Arts    |         3000.00 |
| Akash |  450.00 | NMed    |         5400.00 |
| Bobby |  500.00 | NMed    |         6000.00 |
| Abha  |  300.00 | Comm.   |         3600.00 |
+-------+---------+---------+-----------------+
10 rows in set (0.00 sec)
```

7. To count the number of students with Grade = “A”.

```
mysql> select count(Name) from Student where Grade = 'A';
+-------------+
| count(Name) |
+-------------+
|           4 |
+-------------+
1 row in set (0.00 sec)
```

8. To find the max marks obtained by the student in each grade.

```
mysql> select Grade, max(AvgMark) from Student group by Grade;
+-------+--------------+
| Grade | max(AvgMark) |
+-------+--------------+
| B     |         78.5 |
| A     |         92.0 |
| C     |         68.6 |
+-------+--------------+
3 rows in set (0.00 sec)
```

9. To count the number of students in each class order by class.

```
mysql> select count(Name) Students, Class from Student group by Class;
+----------+-------+
| Students | Class |
+----------+-------+
|        2 | 12B   |
|        1 | 11C   |
|        2 | 12C   |
|        1 | 12D   |
|        2 | 11A   |
|        2 | 12A   |
+----------+-------+
6 rows in set (0.00 sec)
```

10. To count the number of students in each stream who has got C grade.

```
mysql> select Stream, count(Name) C_Grades from Student where Grade = 'C' group by Stream;
+--------+----------+
| Stream | C_Grades |
+--------+----------+
| Comm.  |        2 |
| Arts   |        1 |
+--------+----------+
2 rows in set (0.00 sec)
```