**SECTION 1 – PYTHON**

**Q1. PROGRAM -**

print('Enter No of List Elements :')

list\_num=int(input()) #No of Elements for list

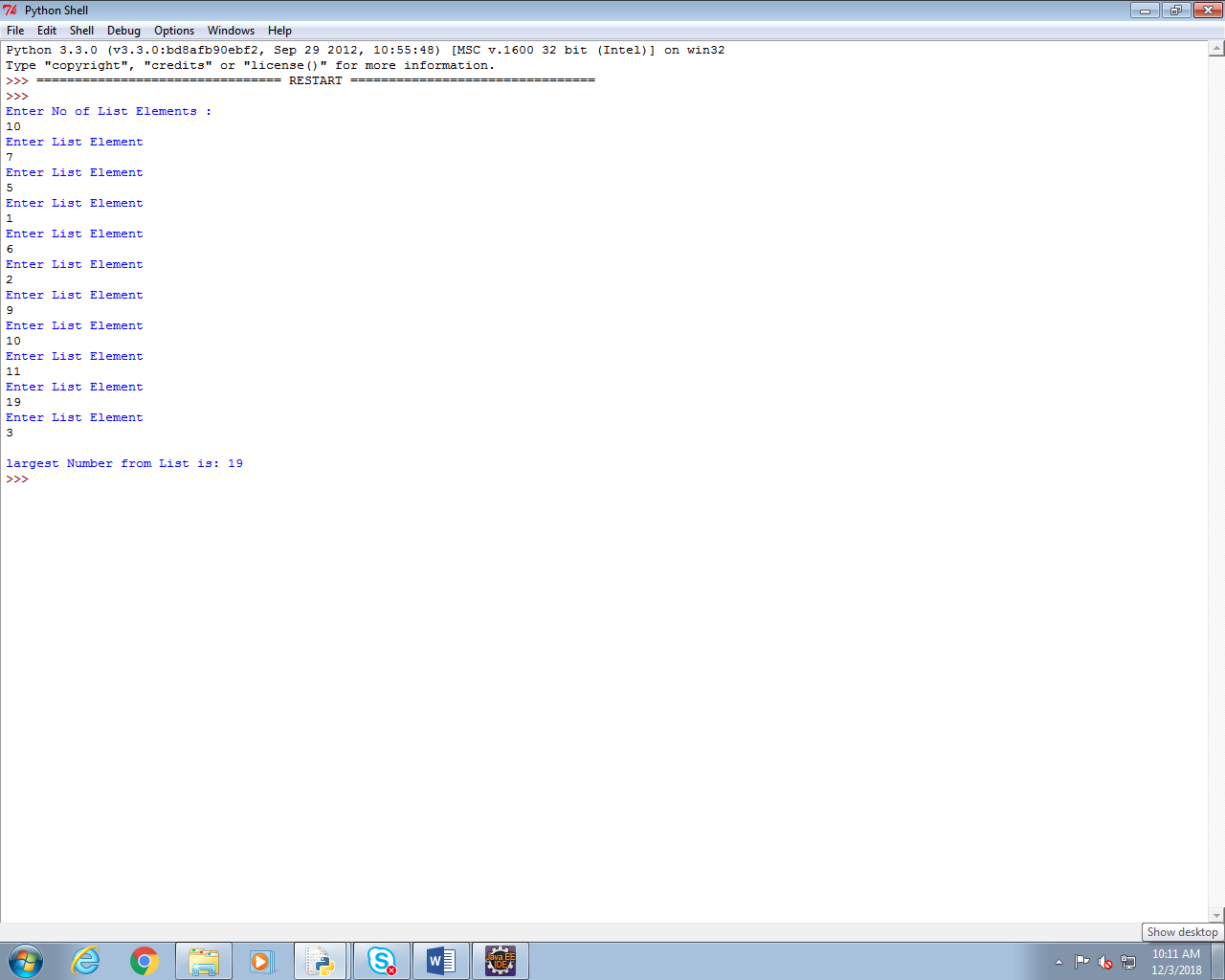
lst=[]

for i in range(list\_num): #Loop to insert data in List

print('Enter List Element')

lst.append(int(input()))

print('\nlargest Number from List is:',max(lst)) #Printing Largest number using max()

**OUTPUT –**

**Q2.**

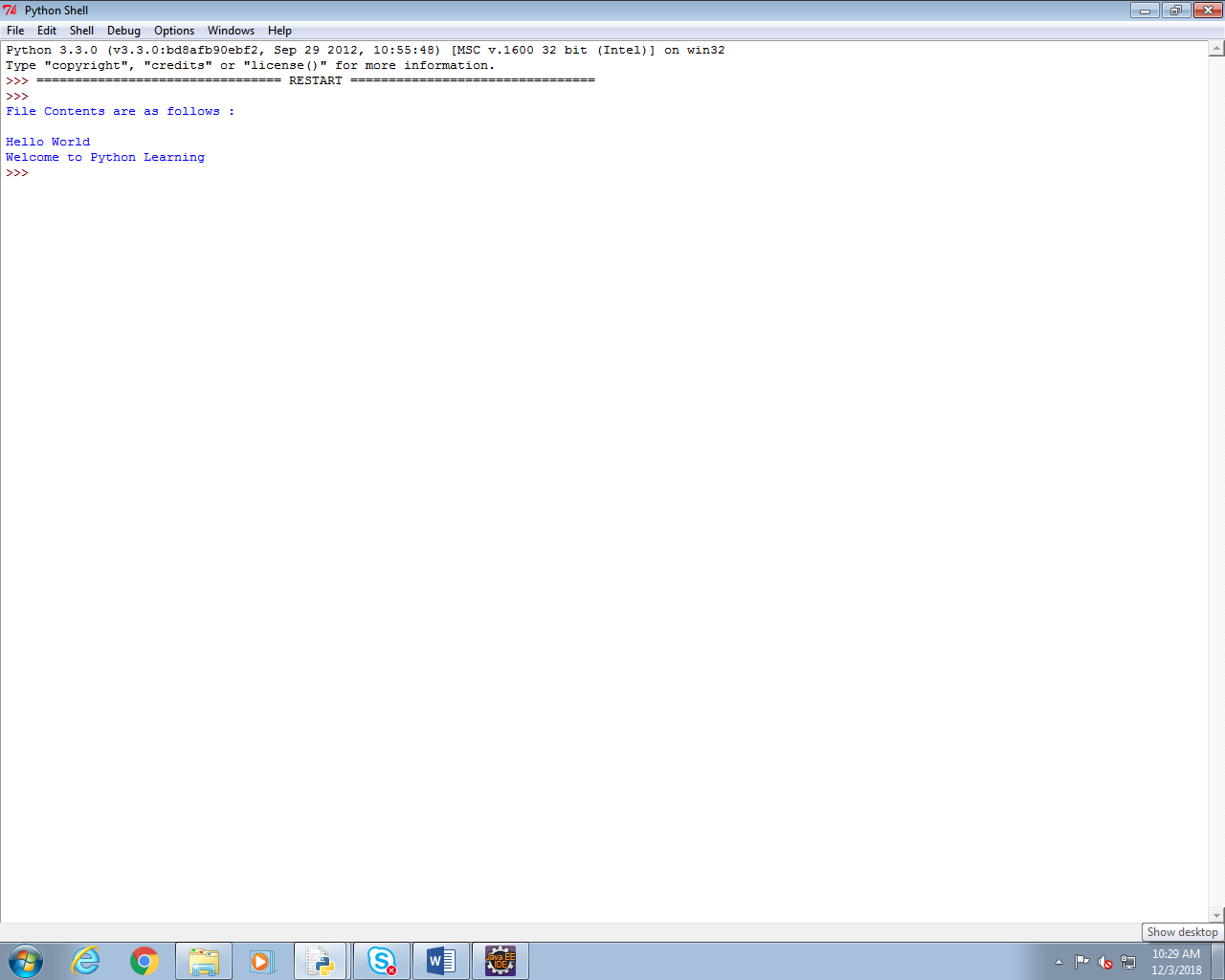
**PROGRAM –**

file = open("sample.txt","r+") #Opening of File and passing mode

print('File Contents are as follows :\n')

print(file.read(),end="") #Reading the File and printing the same

**OUTPUT-**



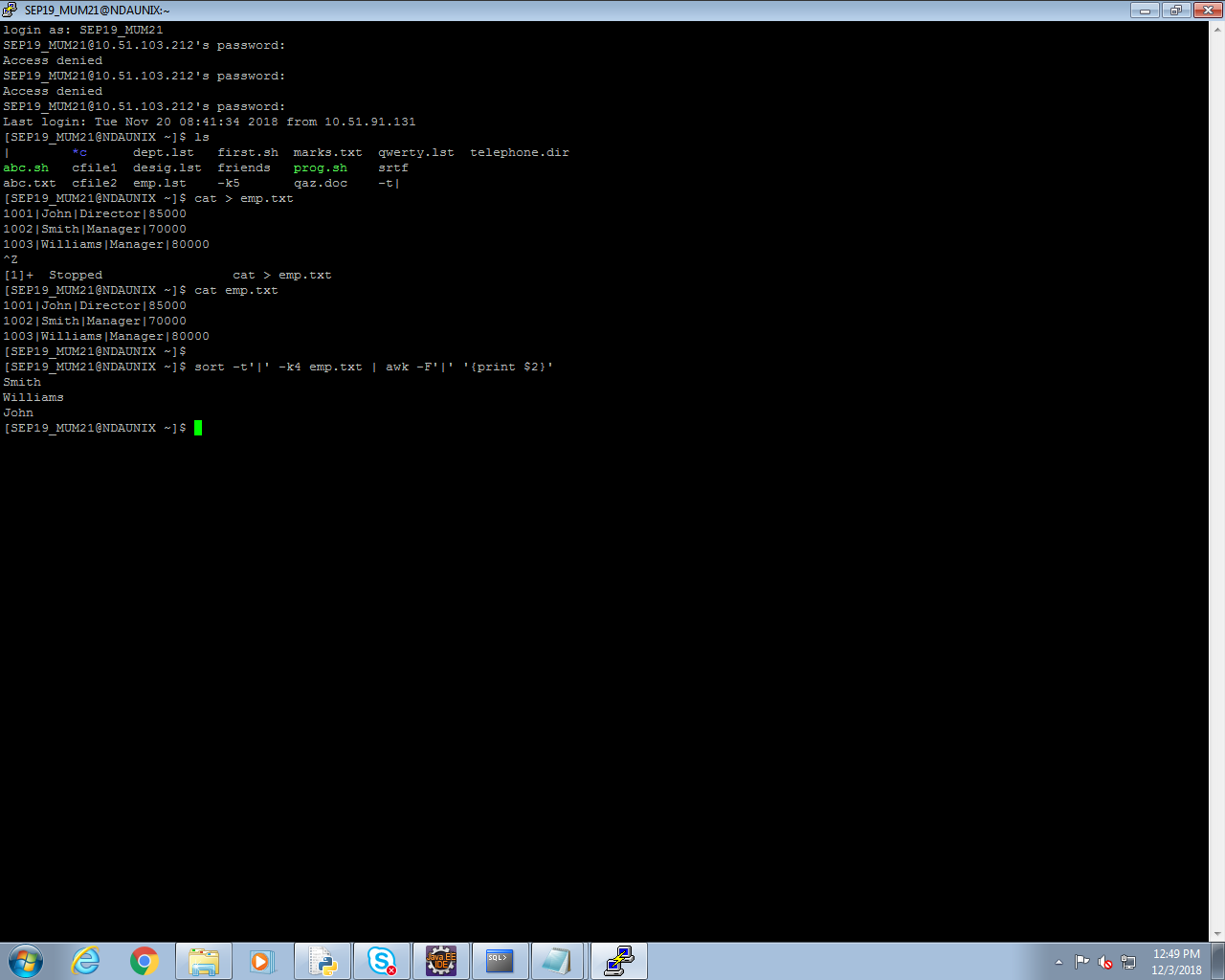
**SECTION 2 – UNIX**

**Q1.**

**1. Command-**

sort -t'|' -k4 emp.txt | awk -F'|' '{print $2}'

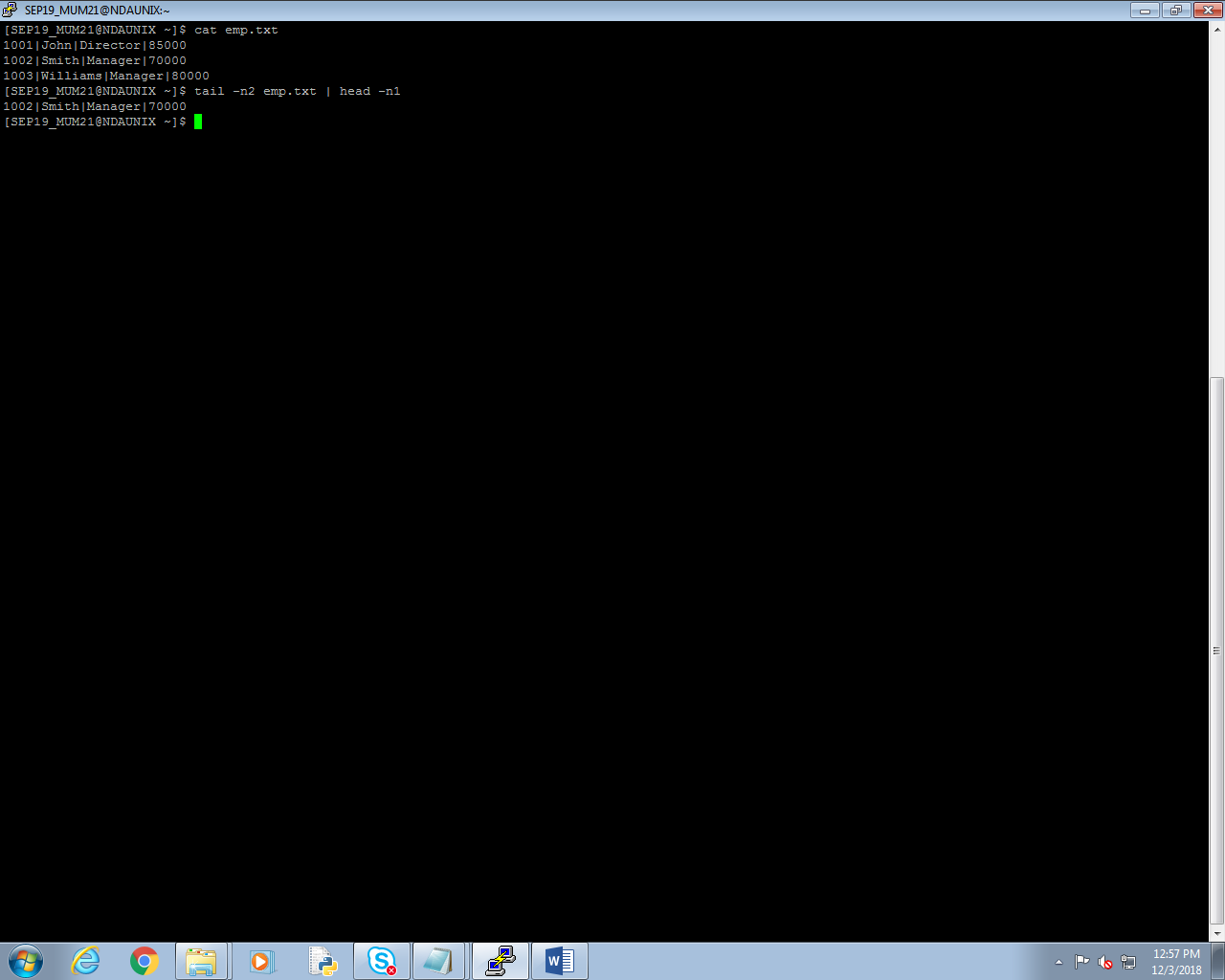
**OUTPUT-**



**2.COMMAND –**

tail -n2 emp.txt | head -n1

**OUTPUT-**



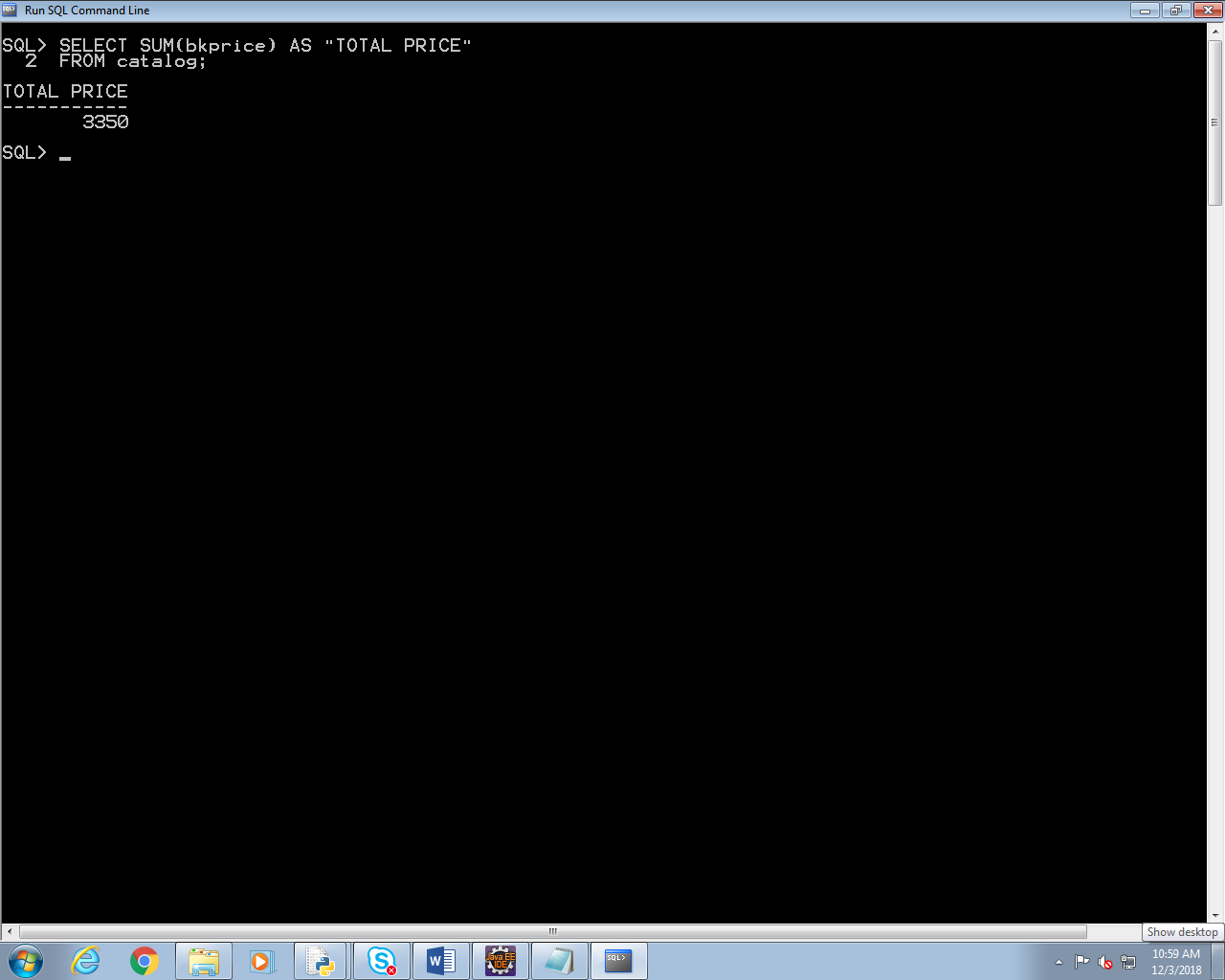
**SECTION 3 – ORACLE**

**Q1.**

1. **QUERY –**

select sum(bkprice) as "Total Price" from Catalog;

**OUTPUT –**



1. **QUERY –**

SELECT title AS "BOOK TITLE"

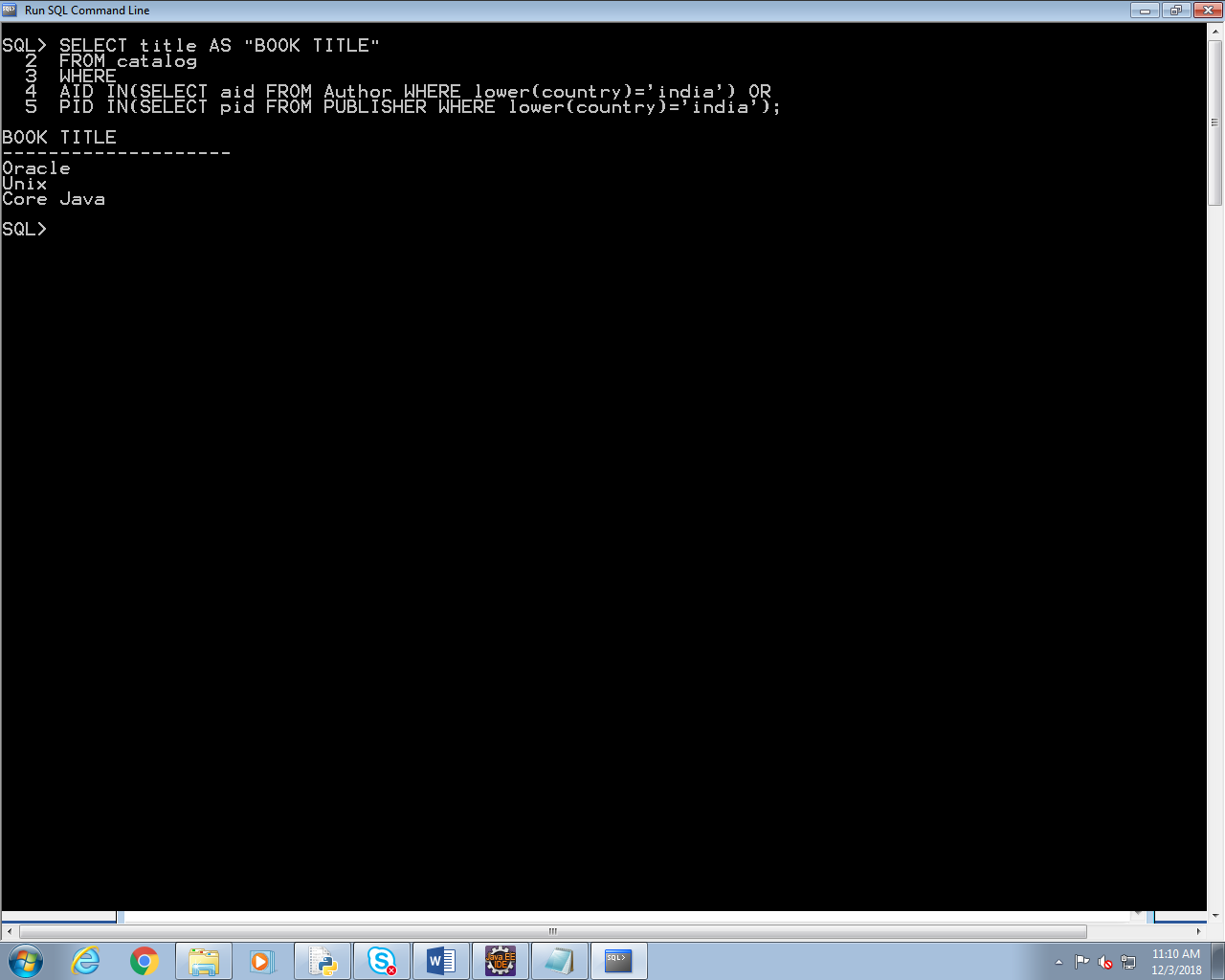
FROM catalog

WHERE

AID IN(SELECT aid FROM Author WHERE lower(country)='india') OR

PID IN(SELECT pid FROM PUBLISHER WHERE lower(country)='india');

**OUTPUT -**

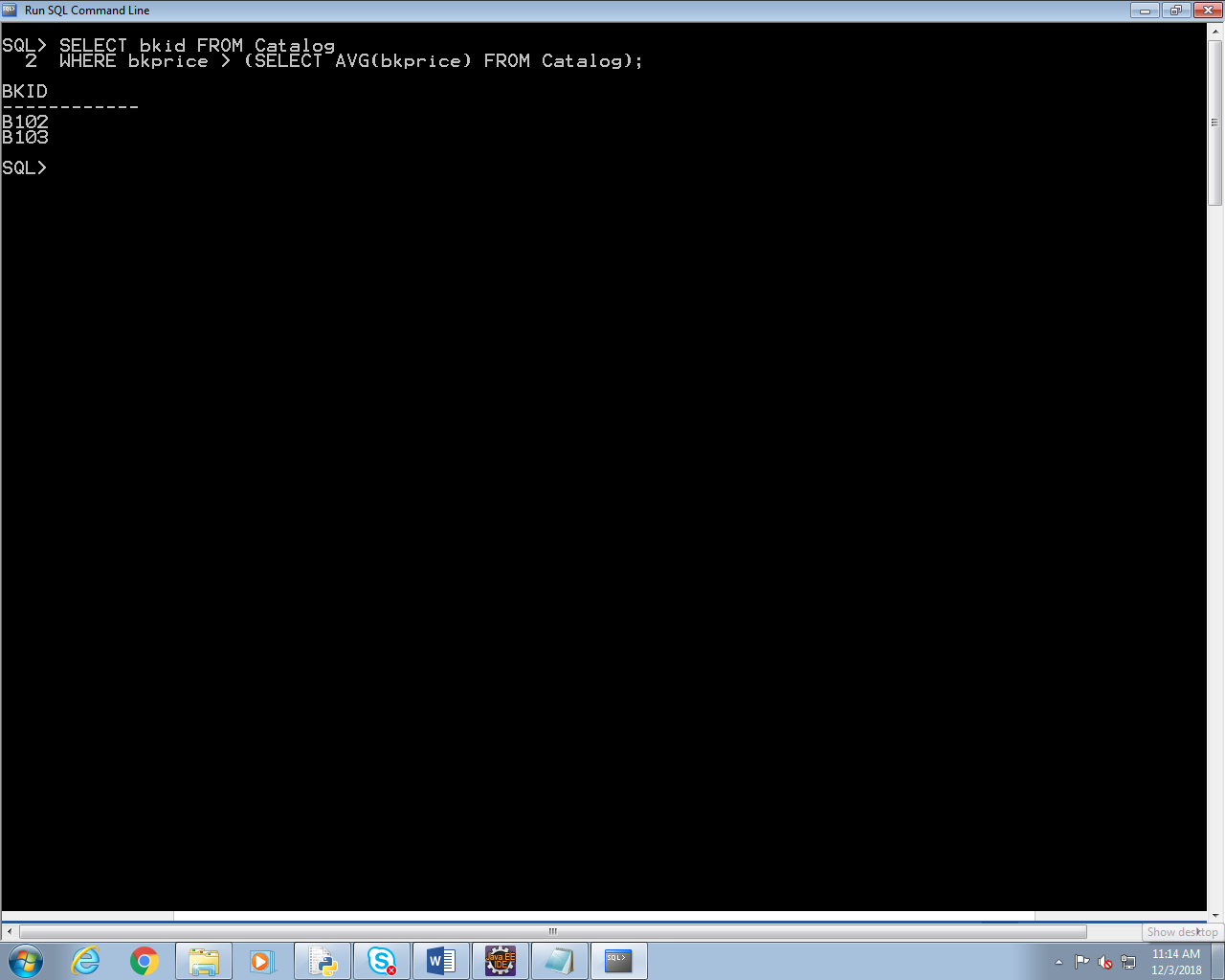


1. **QUERY –**

SELECT bkid FROM Catalog

WHERE bkprice > (SELECT AVG(bkprice) FROM Catalog);

**OUTPUT –**



4.

**QUERY –**

SELECT title FROM

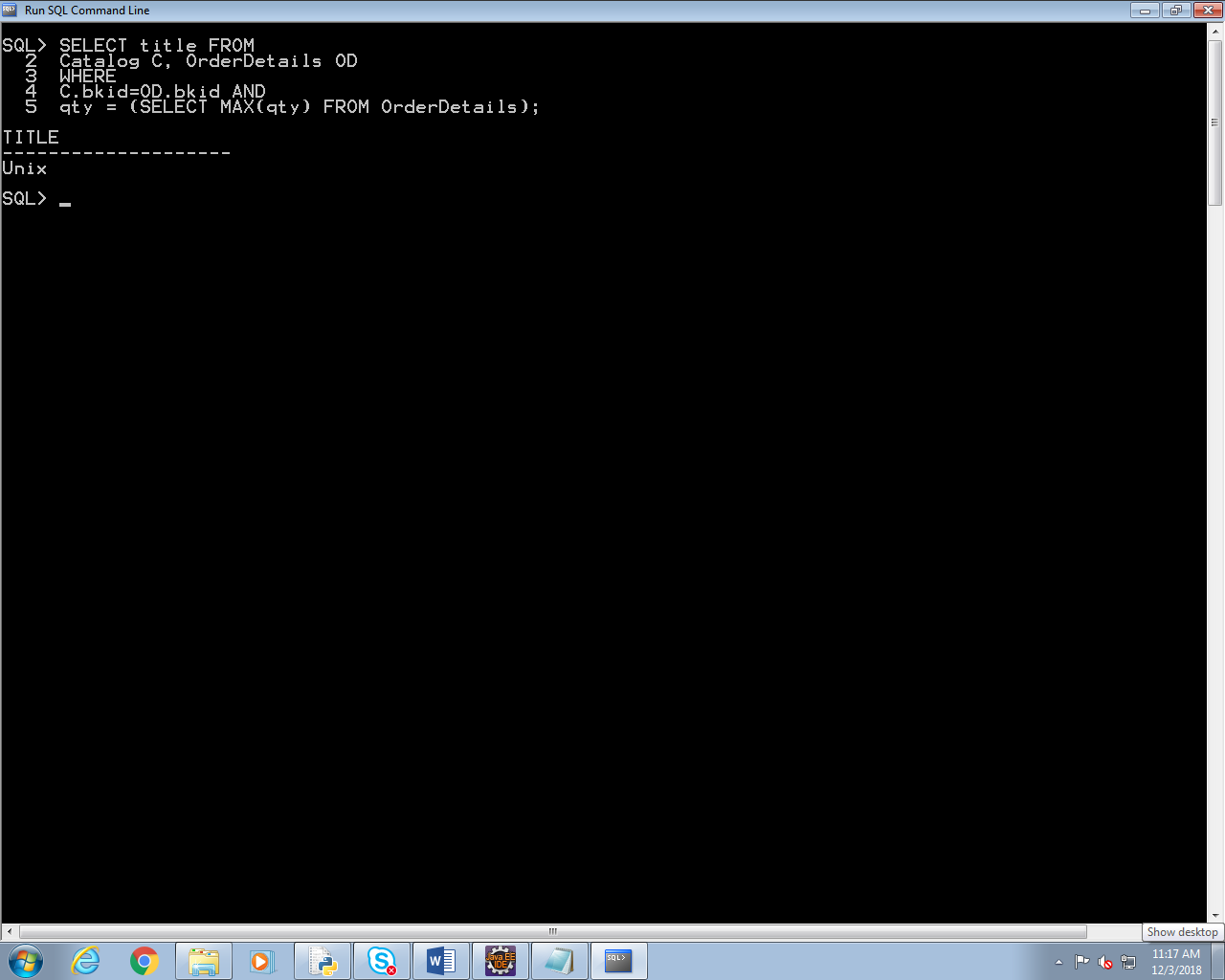
Catalog C, OrderDetails OD

WHERE

C.bkid=OD.bkid AND

qty = (SELECT MAX(qty) FROM OrderDetails);

**OUTPUT –**



**Q2.**

**PROGRAM** -

CREATE OR REPLACE PROCEDURE proc\_update\_price

AS

BEGIN

--Updating Book Price whose Book Price is less than 2000

UPDATE catalog SET bkprice=bkprice\*1.10 Where bkprice<=2000 and lower(cid)='c2';

--Updating Book Price whose Book Price is more than 2000

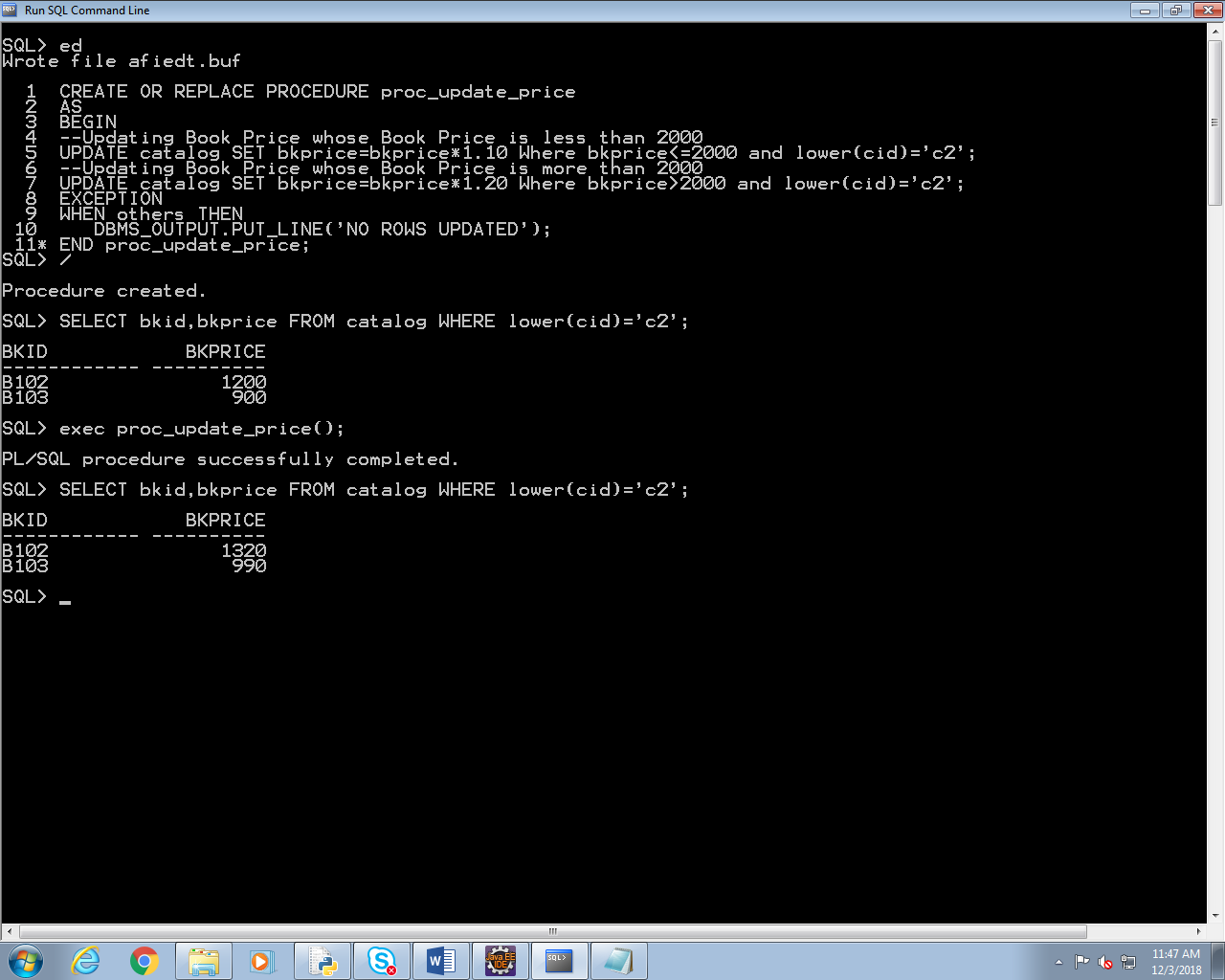
UPDATE catalog SET bkprice=bkprice\*1.20 Where bkprice>2000 and lower(cid)='c2';

EXCEPTION

WHEN others THEN

DBMS\_OUTPUT.PUT\_LINE('NO ROWS UPDATED');

END proc\_update\_price;

**OUTPUT –**

**There is no record with Book Price greater than 2000. So there will be no updation for price greater than 2000.**

**Q3.**

**PROGRAM-**

CREATE OR REPLACE FUNCTION func\_bkid\_update\_price(Book\_Id Catalog.bkid%Type)

RETURN number

AS

var\_edition\_num number;

new\_price number;

Less\_Edition\_exec EXCEPTION;

BEGIN

SELECT edition INTO var\_edition\_num from Catalog WHERE bkid=Book\_Id;

IF var\_edition\_num >3 THEN

--Updating Book Price whose Book Price if edition greater than 3

UPDATE catalog SET bkprice=bkprice\*1.03 Where bkid=Book\_Id;

SELECT bkprice INTO new\_price from Catalog WHERE bkid=Book\_Id;

RETURN new\_price;

ELSE

RAISE Less\_Edition\_exec;

END IF;

EXCEPTION

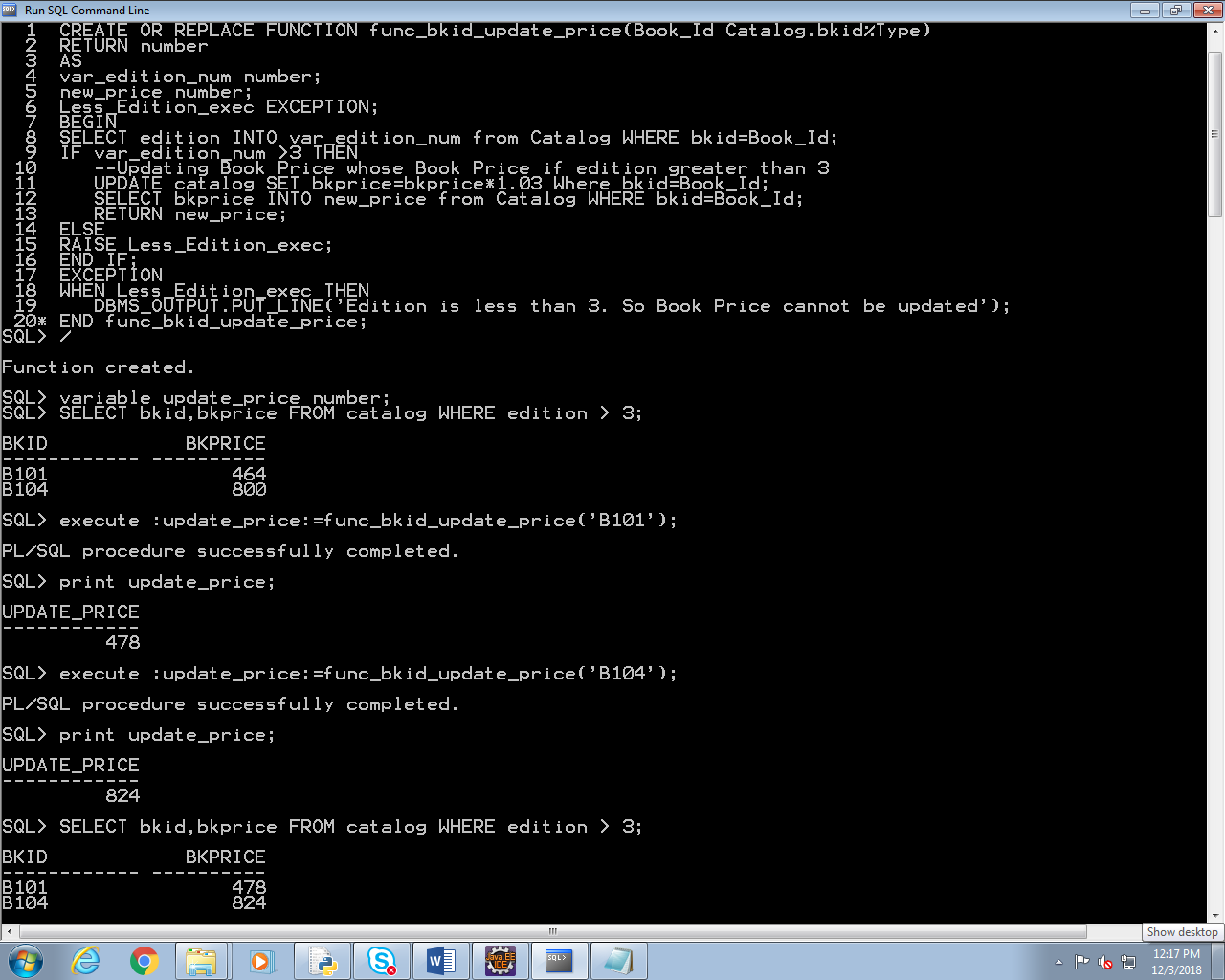
WHEN Less\_Edition\_exec THEN

DBMS\_OUTPUT.PUT\_LINE('Edition is less than 3. So Book Price cannot be updated');

END func\_bkid\_update\_price;

/

**OUTPUT-**



**Q5.**

**QUERY-**

COLUMN ordno HEADING 'ORDER|NUMBER';

COLUMN bkid HEADING 'BOOK|ID';

COLUMN qty HEADING 'QUANTITY|';

COLUMN bkid FORMAT A20;

COLUMN ordn FORMAT A20;

COLUMN "TOTAL PRICE" HEADING 'TOTAL|PRICE';

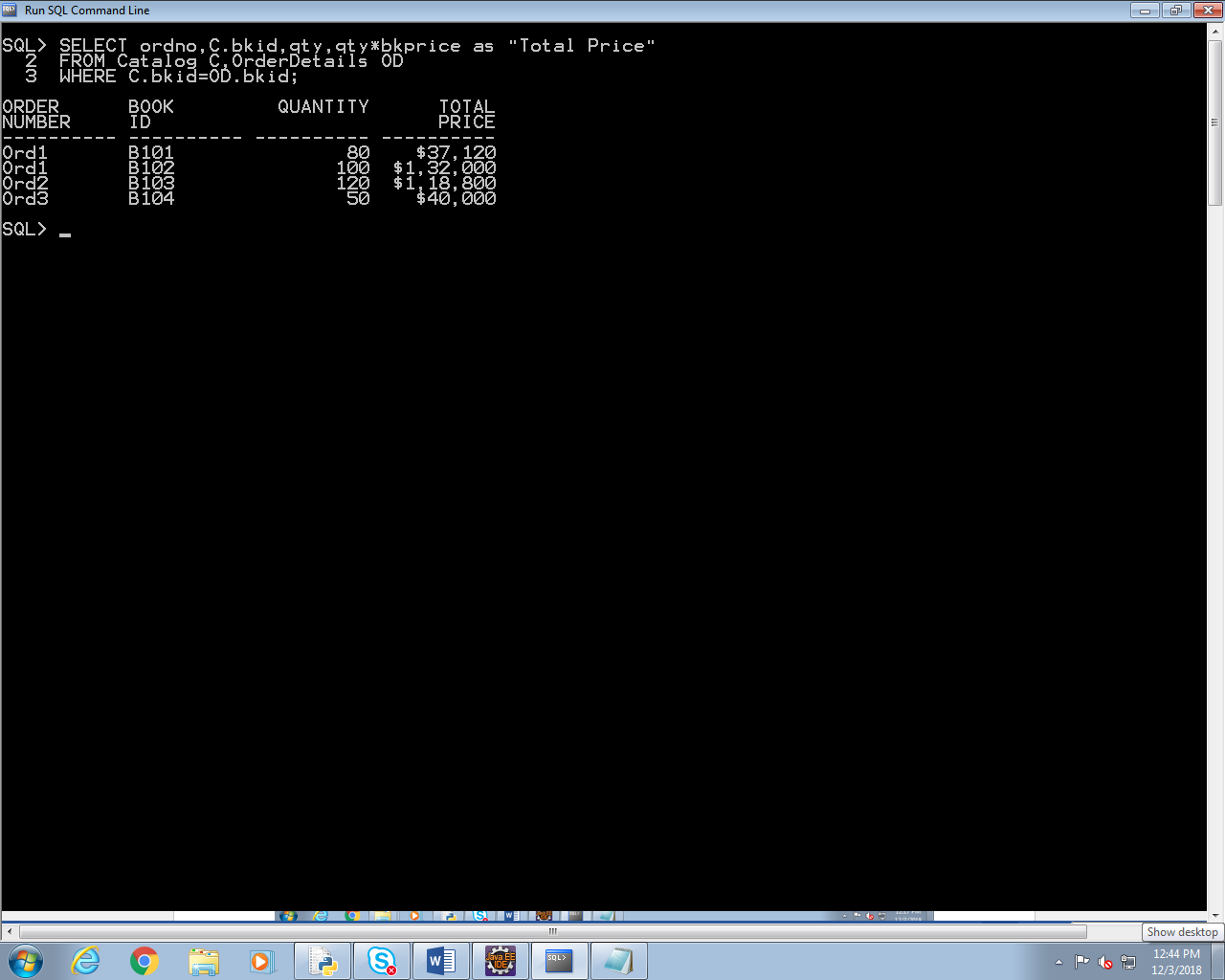
COLUMN "TOTAL PRICE" FORMAT $9,99,999;

SELECT ordno,C.bkid,qty,qty\*bkprice as "Total Price"

FROM Catalog C,OrderDetails OD

WHERE C.bkid=OD.bkid;

**OUTPUT –**



**SECTION 4 – JAVA PROGRAMMING**

**Q1.**

**PROGRAM -**

import java.util.Scanner;

**public** **class** Assessment\_Assign {

**public** **static** **void** main(String[] args)

{

**int** Arr\_size,Arr\_element,i,j,temp;

Scanner sc=**new** Scanner(System.***in***); //Scanner Declaration for I/O

System.***out***.println("Enter Array Length : "); //Array Length

Arr\_size=sc.nextInt();

**int**[] Array=**new** **int**[Arr\_size]; //Array Declaration with given Array size

**for**(i=0;i<Arr\_size;i++)

{

System.***out***.println("Enter Array Element : "); //Entering Array Elements

Arr\_element=sc.nextInt();

Array[i]=Arr\_element;

}

//Sorting of Array in desc using Bubble Sort

**for** (i = 0; i < Arr\_size-1; i++)

{

**for** (j = 0; j < Arr\_size-i-1; j++)

**if** (Array[j] > Array[j+1])

{ //swapping of Array elements

temp=Array[j];

Array[j]=Array[j+1];

Array[j+1]=temp;

}

}

//Printing Second Smallest number with index no 1

System.***out***.println("\n\nSecond Smallest Number :"+Array[1]);

}

}

**OUTPUT –**

