Tshwane University of Technology

We empower people

YEAR: 2012

EXAMINATION: June Supplementary

SUBJECT NAME:	DEVELOPMENT SOFTWARE II/SOFTWARE SKILLS II
SUBJECT CODE:	DSO23BT/SFW20BT
QUALIFICATION(S):	

PAPER DESCRIPTION: Closed Book **DURATION:** 4 Hrs **PAPER:** Only

SPECIAL REQUIREMENTS

- ☐ **NONE**
- ☐ **NON-PROGRAMMABLE POCKET CALCULATOR**
- ☐ **SCIENTIFIC CALCULATOR**
- ☐ **COMPUTER ANSWER SHEET**
- ☐ **GRAPH PAPER**
- ☐ **DRAWING INSTRUMENTS**

OTHER: COMPUTER LAB

INSTRUCTIONS TO CANDIDATES: Answer all questions

TOTAL NUMBER OF PAGES INCLUDING COVER PAGE: 7

TOTAL NUMBER OF ANNEXURES:

EXAMINER: PATRICK MUKALA **FULL MARKS:** 105

MODERATOR: KLAAS MOGAPI **TOTAL MARKS:** 105

Question 1 /6/

A program is required to receive as parameter, the length and width of a room and calculate the area of the room. Determine the price of wall-to-wall carpet if the price of the carpet is R12,58 per square meters. Write the measurement of the room and the price to pay.

Question 2 /6/

Create a programming block that will print the current time and date to the user.

```
G_MESSAGE
------------------------------------------------------------
Today is Friday, 1st June 2012 and it is now 10:33:23
```

Question 3 /7/

Write a block of code that displays the multiples of 10, starting from 110 up until, and including, 100. Example output:

```
Multiples of 10:  10  20  30  40  50  60  70  80  90  100  110
```

Question 4 /12/

Kaleidoscope Bookshop decided that they are going to have a sale. Create a PL/SQL block to decrease RETAIL price of all their books by 10%. Use a cursor with a FOR UPDATE clause to decrease and display the new retail prices accordingly, i.e.

```
The new price for 1059831198 is 22.56
The new price for 0401140733 is 16.04
The new price for 4981341710 is 43.7
The new price for 8843172113 is 40.79
The new price for 3437212490 is 14.54
```

Question 5 /21/

Write a PL/SQL block to retrieve books from the BK_BOOKS table based on a PUBID supplied interactively at run time by the user. Use appropriate non-redefined exceptions (using PRAGMA EXCEPTION_INIT) as define in the table below to handle the errors and display the message as indicated otherwise display the book title, the author name (lastname and first name separated by commas), the publisher name and the book classification.

TYPE OF ERROR	MESSAGE
Unspecified error	Unknown error
Returns no books	No books from this publisher
Returns more than one rows	More books from the publisher

Question 6 /20/

Kaleidoscope Bookshop need to know, from time to time, how many copies of a certain book has been sold.

- Declare a table of type NUMBER(4).
- Declare a cursor that receives one parameter (the ISBN number as supplied by the user). The cursor must retrieve all records from the BK_ORDERITEMS table with this ISBN number.
- Using a BASIC LOOP, read through the cursor, storing the QUANTITY sold in consecutive elements of the table.
- Use a counter to determine the number of items stored in the table.
- Then use a FOR LOOP to read through the table, finding the sum of all the elements. Display the sum.
- Example:

```
Enter value for ISBN:  8843172113
Old 12:  OPEN CUR_ITEMS(&ISBN);
New 12: OPEN CUR_ITEMS(8843172113);
We sold:  7 of the book
PL/SQL procedure successfully completed
```

Question 7 **/18/**

Execute the script file **ALTER_BK_BOOKS.SQL** provided to you by your lecturer. This script file will alter the **BK_BOOKS** table by adding a **QUANTITY** column to the table and initialising all quantities with the value **10**. Content of the script file

ALTER_BK_BOOKS.SQL:

```
alter table bk_books
 add quantity number(3) default 10;
commit;
```

SQL> Select * from bk_books;

ISBN	TITLE	PUBDATE	PUBID	CATEGORY	AUTH	QUANTITY
1059831198	BODYBUILD IN 10 MINUTES A DAY	21-JAN-01	4	FITNESS	S100	10
0401140733	REVENGE OF MICKEY	14-DEC-01	1	FAMILY LIFE	J100	10
4981341710	BUILDING A CAR WITH TOOTHPICKS	18-MAR-02	2	CHILDREN	K100	10
8843172113	DATABASE IMPLEMENTATION	04-JUN-99	3	COMPUTER	P105	10
3437212490	COOKING WITH MUSHROOMS	28-FEB-00	4	COOKING	B100	10
3957136468	HOLY GRAIL OF ORACLE	31-DEC-01	3	COMPUTER	A100	10
1915762492	HANDCRANKED COMPUTERS	21-JAN-01	3	COMPUTER	W105	10
9959789321	E-BUSINESS THE EASY WAY	01-MAR-02	2	COMPUTER	J100	10
2491748320	PAINLESS CHILD-REARING	17-JUL-00	5	FAMILY LIFE	R100	10
0299282519	THE WOK WAY TO COOK	11-SEP-00	4	COOKING	S100	10
8117949391	BIG BEAR AND LITTLE DOVE	08-NOV-01	5	CHILDREN	R100	10
0132149871	HOW TO GET FASTER PIZZA	11-NOV-02	4	SELF HELP	S100	10
9247381001	HOW TO MANAGE THE MANAGER	09-MAY-99	1	BUSINESS	W100	10
2147428890	SHORTEST POEMS	01-MAY-01	5	LITERATURE	W105	10

14 rows selected.

Contents of BK_BOOKS after ALTER_BK_BOOKS has been executed

(*Note that the Quantity column has been updated to 10*)

The Bookstore's Management team requires information regarding all books that has been ordered according to a specific order number (**P_ORDER#**) supplied as an input parameter.

Your job is to create a Server side procedure **Show_Books** that will accept **a single** parameter **P_ORDER#** at runtime, for e.g.:

P_ORDER

SQL> execute show_books(1000);

```
Order#      Item#      TITLE OF THE BOOK                  #ORDERED
======      =====      ==================                 ========
1000        1          COOKING WITH MUSHROOMS             1

PL/SQL procedure successfully completed.
```

Procedure Show_Books executed using order# 1000

The following criteria should be followed:

Create an implicit cursor **Book_Cur** that will return a resultset that contains the ORDER#, ITEM#, ISBN, TITLE and QUANTITY fields. Use a **Cursor For Loop** to print the resultset returned by the cursor. ***Refer to the output listed above. Take note of the headings used as well as any special formatting used, as marks shall be allocated for this.***

Question 8 /10/

Create a Server side function **GET_ITEM_NO** that will accept **a SINGLE** parameter at runtime **P_ORDER#**.

The following criteria should be followed:

This function must return a value; i.e. the next available item# based on the order# (**P_ORDER#**) passed to the function. ***This fuction is required for specific customer's that may want to add items to an existing order that has already been placed.***

For e.g. if you test your function in SQL Plus using the ORDER# 1001, your function should return the following item#:*

```
SQL> variable v_item# number

SQL> execute :v_item# := get_item_no(1001);

PL/SQL procedure successfully completed.

SQL> print v_item#

V_ITEM#
---------
3
```

Function GET_ITEM_NO invoked in SQL*Plus using the bind variable v_item#

The price **v_item#** is populated during execution of the function and returned to the calling environment. You may assume that the function will always receive a valid order number. Hence, you don't need to make provision for any error handling techniques.

Use the ***example above*** to define a suitable variable to test your function.