UNIVERSITY EXAMINATIONS

UNIVERSITEITSEKSAMENS

UNISA university of south africa

INF2611

October/November 2017

VISUAL PROGRAMMING II

Duration 2 Hours

70 Marks

EXAMINERS

FIRST	MS E LEUS
SECOND	MRS A MATHEW

Closed book examination

This examination paper consists of 14 pages

The examination paper is divided into two sections, namely section A and section B. Please answer only one section.

- Section A, which covers Python, is for students who registered for the subject in 2017.
- Section B, which covers Delphi, is for students who were registered for the subject prior to 2017 and are writing a supplementary or special examination

Instructions

- Answer all the questions in the answer book
- Answers in pencil will not be marked
- The marks are provided in brackets next to the questions
- Enjoy!

Duration. 2 hours
Marks 70

[Turn over]

Section A: Python

Question 1 - Advanced Widgets and Menus (30)

1 1 Displaying LCD digits

a) Explain how to display LCD-like widgets, by referring to the relevant widget and class names (2)

b) Which method returns the numerical value displayed by the widget referred to in question 1 1 a? (1)

c) What is the purpose of the `setMode()` method in displaying LCD digits? Provide an example (2)

1 2 Timers.

a) Explain the use of timers in Python and how to apply them in an application (3)

b) List and explain two methods that control the `timeout()` signal (4)

1 3 Calendar widget
Consider the following code and answer the questions that follow

```
   #callcalendar.pyw
1     import sys
2.    from dispcalendar import *
3.
4.    class MyForm(QtGui.QDialog):
5.       def __init__(self, parent=None):
6.          QtGui.QWidget.__init__(self, parent)
7           self ui = Ui_Dialog()
8           self ui.setupUi(self)
9           QtCore.QObject.connect(self.ui.calendarWidget,
10.         QtCore.SIGNAL('selectionChanged()'),
11          self dispdate)
12.
13       def dispdate(self)
14          self ui dateEdit.setDate
15.         (self.ui calendarWidget.selectedDate())
16.
17.   if __name__ == "__main__":
18.      app = QtGui.QApplication(sys argv)
19.      myapp = MyForm()
20       myapp.show()
21       sys.exit(app exec_())
```

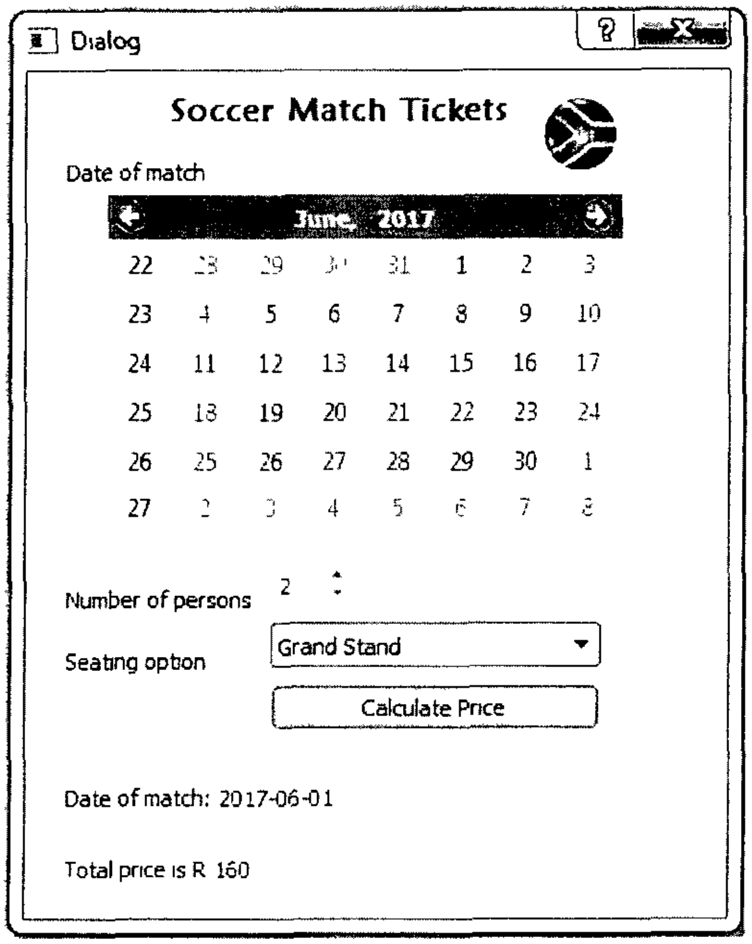
a) Explain the purpose of the `selectionChanged()` signal in line 10 by referring to the function it is connected to (2)

b) Provide the name of the method that retrieves the date selected by the user and the name of the widget that displays the output (2)

c) The output of the code above provides the date in the following format `2017/05/18`
Provide an additional line of code that will display the date in the following format
`18 May 2017` (2)

1 4 Combo Box widget
The following program calculates the price for the purchase of soccer match tickets by asking the date of the match, the number of persons attending and the seating option the user prefers The Combo Box will display four seating options VIP, Grand Stand, East Stand and Open Wing The prices for the seating options are

VIP R200
Grand Stand R80
East Stand R60
Open Wing R40



Date of match: 2017-06-01

Total price is R 160

Figure 1

```
#ticketprice.pyw
import sys
from soccermatch import *

class MyForm(QtGui.QDialog):
    def __init__(self, parent=None):
        QtGui.QWidget.__init__(self, parent)
        self.ui = Ui_Dialog()
        self.ui.setupUi(self)
        self.seatingoptions=['VIP', 'Grand Stand', 'East Stand', 'Open
  Wing' ]
        self.addcontent()
        QtCore.QObject.connect(self.ui.pushButton,
  QtCore.SIGNAL('clicked()'), self.
        calculateprice)
    def addcontent(self)
        for i in self.seatingoptions:
            self.ui.comboBox.addItem(i)

    def calculateprice(self):
        dateselected=self.ui.calendarWidget.selectedDate()
        dateinstring=str(dateselected.toPyDate())
        noOfPersons=self.ui.spinBox.value()
        chosenoption=self.ui.comboBox.itemText
        (self.ui.comboBox.currentIndex())
        self.ui.Enteredinfo.setText('Date of match:
        '+dateinstring)

if __name__ == "__main__":
    app = QtGui.QApplication(sys.argv)
    myapp = MyForm()
    myapp.show()
    sys.exit(app.exec_())
```

a) Provide the missing code that will compute and display the price of the tickets purchased based on the number of people attending and seating options selected

(7)

1.5. Menus

a) Explain two methods used to add a menu entry to a menu (4)

b) Explain the use of the `statusTip` property (1)

Question 2 - Multiple Documents and Layouts (10)

Consider the following application and answer the questions that follow

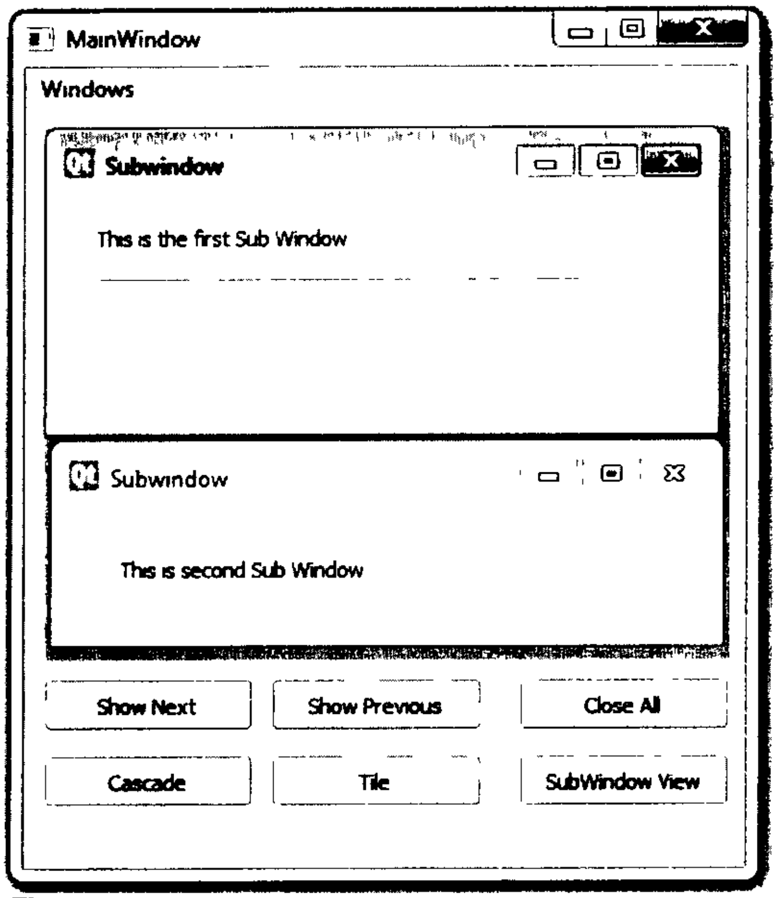


Figure 2

```
#callMDI pyw
import sys
from mdi import *
class MyForm(QtGui QMainWindow)
   def __init__(self, parent=None)
       QtGui QWidget __init__(self, parent)
       self.ui = Ui_MainWindow()
       self.ui setupUi(self)
       self ui mdiArea addSubWindow(self.ui subwindow)
       self ui.mdiArea.addSubWindow
       (self.ui.subwindow_2)
       QtCore QObject.connect(self.ui showNext,
        QtCore.SIGNAL('clicked()'), self.displayNext)
        QtCore.QObject.connect(self.ui.showPrevious,
        QtCore.SIGNAL('clicked()'),
        self.displayPrevious)
        QtCore.QObject connect(self.ui.closeAll,
        QtCore.SIGNAL('clicked()'), self.closeAll)
        QtCore.QObject.connect(self.ui cascadeButton,
        QtCore.SIGNAL('clicked()'),
        self.cascadeArrange)
        QtCore.QObject.connect(self.ui.tileButton,
        QtCore.SIGNAL('clicked()' ), self.tileArrange)
        QtCore.QObject.connect(self.ui.
        SubWindowViewButton,
        QtCore.SIGNAL('clicked()'), self SubWindowView)
        self.connect(self.ui.actionFirst_Window,
        QtCore.SIGNAL('triggered()' ),
        self displayNext)
        self.connect(self.ui.actionSecond_Window,
        QtCore.SIGNAL('triggered()'),
        self.displayPrevious)
1.______________________________________________
    def displayNext(self):
        self.ui.mdiArea.activateNextSubWindow()
2.______________________________________________
    def displayPrevious(self):
        self.ui.mdiArea.activatePreviousSubWindow()
3.______________________________________________
    def closeAll(self):
        self.ui.mdiArea.closeAllSubWindows()
4.______________________________________________
    def cascadeArrange(self):
        self.ui.mdiArea.cascadeSubWindows()
5.______________________________________________
    def tileArrange(self):
        self.ui.mdiArea.tileSubWindows()
6.______________________________________________
    def SubWindowView(self):
        self.ui.mdiArea.setViewMode(0)

if __name__ == "__main__":
    app = QtGui.QApplication(sys.argv)
    myapp = MyForm()
    myapp.show()
    sys.exit(app.exec_())
```

2.1. Provide the comments numbered 1 to 6 above, which explains the functions of the dispNext, dispPrevious, closeAll, cascadeArrange, tileArrange and SubWindowView buttons in Figure 2 (6)

2 2 List and explain two types of layout managers for widgets in Qt Designer (2)

2 3 List and explain two Group Box properties or methods (2)

Question 3 - Database Handling (15)

3 1 Provide one line of SQL code to create a database called `clinic` at the MySQL prompt (1)

3 2 The Python code for creating a database table called `patients` is as follows

```
createtable.py
import sys
import MySQLdb
conn=MySQLdb connect(host="localhost", user="root", passwd="psw",
db="clinic")
cursor=conn.cursor()
try:
  cursor.execute ("""
  ______________________________________
  ______________________________________
  """)
except MySQLdb Error
  print ("Error in creating patients table")
  sys exit(1)
cursor.close()
conn.close()
```

3 2 1 Provide the missing code to create the `patients` table, which includes the following fields (4)
`patient_id smallint`, should not be '0'
`patient_name`, should not exceed 40 characters
`patient_balance, float`

3 2 2 Explain the use of the `cursor()` method (2)

3 2 3. Explain the use of the `execute()` method (2)

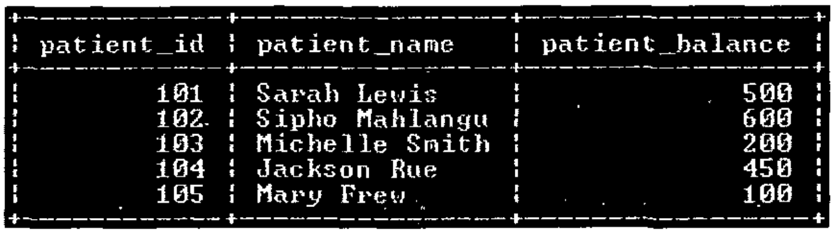
3 3 Explain how you will connect your application to the database server by referring to the relevant method and parameters (3)

3 4 Provide three lines of SQL code necessary to display 1) *all the tables in the* `clinic` database, 2) the structure of the `patients` table and 3) the records/rows in the `patients` table at the MySQL prompt (3)

Question 4 - Console-based Database Maintenance (15)

The following table was created in the database called `clinic`

```
+------------------+
| Tables_in_clinic |
+------------------+
| patients         |
+------------------+
```



```
+------------+---------------+-----------------+
| patient_id | patient_name  | patient_balance |
+------------+---------------+-----------------+
|        101 | Sarah Lewis   |             500 |
|        102 | Sipho Mahlangu|             600 |
|        103 | Michelle Smith|             200 |
|        104 | Jackson Rue   |             450 |
|        105 | Mary Frew     |             100 |
+------------+---------------+-----------------+
```

4 1 Provide the missing code, which will delete the given record from the `patients` table and print a message that states that the record was deleted from the table The code should also prompt the user to confirm the deletion of the record by indicating Yes/No before deleting it and print an appropriate message if the record that the user is requesting to delete cannot be found (5)

patient_isbn	patient_name		patient_balance
105	Mary Frew		100

```
#sqldelete.py
import pymysql
conn=pymysql connect(host="localhost", user="root", passwd="psw",
db="clinic")
cursor=conn cursor()
p=int(input("Enter Patient ID  "))
cursor execute ("SELECT * from patients where patient_id=%d" %p)
row=cursor fetchone()

_______________________________
_______________________________
_______________________________
_______________________________
_______________________________

cursor.close()
conn.commit()
conn close()

Output
Patient with ID 105 is deleted
```

4 2 Provide comments for the followıng sectıon of code, whıch explaıns the steps taken by the Python program to fetch rows from the `patıents` table (10)
Note- the comment should refer to the code dırectly below the comment.

```
#dısprec1.py
import sys
ımport pymysql
1. ______________________________________________
conn=pymysql.connect(host="localhost", user="root", passwd="psw",
db="clinic")
cursor=conn cursor()
try·
2. ______________________________________________
    cursor.execute ("SELECT * from patients")
3. ______________________________________________
    print ("Patient ID\tPatient Name\tPatıent Balance")
4. ______________________________________________
    while(1):
5. ______________________________________________
        row=cursor.fetchone()
6. ______________________________________________
        if row==None:
            break
7. ______________________________________________
        prınt ("%d\t\t%s\t\t%d\t\t%f" %(row[0], row[1], row[2],
row[3]))
8 ______________________________________________
except MySQLdb.Error:
    prınt ("Error in fetching rows")
    sys.exit(1)
9. ______________________________________________
cursor.close()
10. _____________________________________________
conn.close()
```

```
Output :
Patıent ID      Patient Name      Patıent Balance
101             Sarah Lewıs       R500
102             Sıpho Mahlangu    R600
103             Mıchelle Smith    R200
104             Jackson Rue       R450
```

Section B: Delphi

Question 1 - Menus (15)

1.1. List and describe three types of menus that can be implemented in a Delphi application (6)

1 2 Give the necessary steps on how to add an item to a menu (2)

1 3 Give four reasons why is it necessary to include menus in an application (4)

1.4 Define two-way centralisation by referring to the use of different sources to trigger the same operation (3)

Question 2 - Data modules and multiple forms (15)

2.1. Define a data module. (2)

2 2 What are the benefits of including a database module in your application? (3)

2 3 The `University` database contains a `Students` table, which provides student records that consists of student numbers, student names and student categories

StudentNumber	**StudentName**	**StudentCategory**
15589	Amanda Smith	1st year
12256	Julius Furrow	2nd year
18897	Shawren Kim	1st year
14489	Blade Lewis	2nd year

2 3.1 Provide the SQL command for retrieving records from the `Students` table from the student category. 1st year, which is ordered according to student names. (3)

2.4. Give the definition of modal forms and provide two examples of these type of forms (3)

2 5 Explain the difference between 'auto-create forms' and 'available forms' Differentiate between the two ways of creating forms by referring to 'auto-create forms' and 'available forms' (4)

Question 3 - Database applications (20)

3 1 Provide the necessary steps required to create and connect to a database through the console manager (3)

3 2. Why is it necessary to connect to a database? (2)

3 3 Describe the DBGrid component and how it can be applied when working with databases in Delphi (3)

3 4 Name the method that is used to apply updates to a database table (1)

[Turn over]

3 5 Name the method that is used to undo recent changes made to a database table (1)

3 6 Explain why it is necessary to use key fields in database tables (2)

3.7 The schema in Figure 1 illustrates the relationships between the components used to retrieve records from a database to the `ClientDataSet` and to the data control components of a form Indicate the missing components by referring to the corresponding letters in Figure 1 Provide the name of each component, its property type and a short description (8)

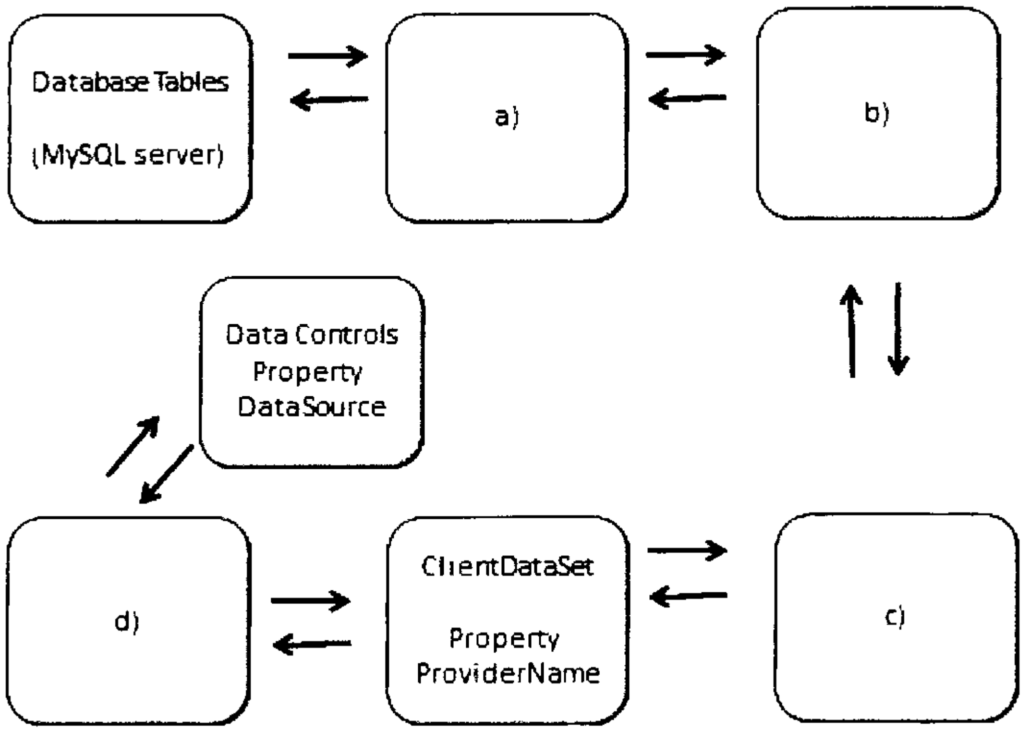


Figure 1

Question 4 – Object orientation (20)

4.1. Figure 2 illustrates a basic unit file in Delphi

```
unit Unit1;

interface

uses
 Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
 Dialogs;

type
 TForm1 = class(TForm)
 Button1: TButton;
 procedure Button1Click(Sender: TObject);
 private
 { Private declarations }
 public
 { Public declarations }
 end;

var
 Form1: TForm1;

implementation

{$R *.dfm}

procedure TForm1.Button1Click(Sender: TObject);
begin

end;

end.
```

Figure 2

4.1.1 Provide an explanation of line 1 in Figure 2. `unit Unit1;` also specify if it appears as a default for Delphi applications or if it may differ from application to application. (2)

4.1.2 Describe the interface section and its subsections (4)

4.1.3 Provide an explanation of the implementation section. (2)

4.2 An object is a self-contained entity that has state and behaviour through its attributes Describe encapsulation (2)

4 3 What ıs the dıfference between an objects and a class? (2)

4 4 Provıde the three steps requıred for creatıng objects (3)

4.5 What ıs the dıfference between a *subclass* and a *superclass*? (2)

4 6. What ıs the motıvatıon behınd usıng ınherıtance for object orıentatıon? (3)