higher education
& training

Department:
Higher Education and Training
REPUBLIC OF SOUTH AFRICA

NATIONAL CERTIFICATE (VOCATIONAL)

COMPUTER PROGRAMMING: PRACTICAL
(Second Paper)
NQF LEVEL 4

NOVEMBER EXAMINATION

(10041024)

6 November 2013 (X-Paper)
09:00–13:00

This question paper consists of 16 pages.

TIME: 4 HOURS
MARKS: 100

INSTRUCTIONS AND INFORMATION

1. Answer ALL the questions.
2. Read ALL the questions carefully.
3. Number the answers according to the numbering system used in this question paper.

INSTRUCTIONS TO STUDENTS AND LECTURERS

1. Students MUST be seated in the computer laboratory 30 minutes before the start of the examination to ensure that ALL computers have the appropriate programs.
2. These questions must be completed on a computer by making use of MS Office 2003 or a later version and VB.NET 2005 or VB.NET Express or VB.NET 2008.
3. Students must make sure that they print their work immediately after completing a question. Ensure that your EXAMINATION NUMBER appears on each printout. NO QUESTIONS WITHOUT A PRINTED EXAMINATION NUMBER WILL BE MARKED.
4. In the event of problems such as: a power failure; computer breakdown or a printer breakdown, the invigilator will make the necessary arrangements to continue with the examination process once the problem has been resolved. Students will not be penalised with time lost under these circumstances.
5. ALL work on the computer should be saved at regular intervals to prevent loss of work. NO additional time may be allowed for work lost due to lack of saving.
6. NO student may print his/her work for another student, or make a memory stick available to another student or access another student's work on the network. Any attempt to access any information from or transfer information to another student in whatever manner is a contravention of the examination rules and regulations and will be viewed in an extremely serious light.
7. ALL printouts or screen prints done during the session MUST be handed in.

8. Make sure that your EXAMINATION NUMBER appears on ALL printouts.

9. Students may NOT copy the source code from VB.NET to MS Word and then print it from the word processor.

10. Steps to print a VB.NET form or any other screen:

    10.1 Run the solution.

    10.2 Hold the ALT key down and press the print screen button (This will copy the active screen).

    10.3 Open MS Word and paste the form/screen in a new document.

    10.4 Add your EXAMINATION NUMBER to the top right-hand side of your page.

    10.5 Print the page.

11. You are allowed to make use of the help files in VB.NET.

12. The following files will appear on your hard drive in the folders:

    12.1 QUESTION 1
    C:\CP L4 NOV 2013\LuxuryCar\LuxuryCar

    12.2 QUESTION 2
    C:\CP L4 NOV 2013\UniqueFitness\UniqueFitness;
    ...\UniqueFitness\UniqueFitness\Bin\Debug\FitnessSystems.mdb

    12.3 QUESTION 3
    C:\CP L4 NOV 2013\YouthDevelopment

    Logo.jpg;
    Content.txt;
    Styles.css

13. Students may use the following or any other program of their choice to create web pages:

- NOTEPAD ++
- NETBEANS
- BLUE VODA
- DREAMWEAVER

QUESTION 1

The following questions refer to the VB.NET solution called *LuxuryCar (C:\CP L4 NOV 2013\LuxuryCar\LuxuryCar)* that has been saved on the hard drive.

Design an application for the company LUXURY LIFE STYLE that will help them to calculate the amount due for the rental of cars, depending on the option the client chooses. A client may choose between a private car and a mini-bus. A client may not hire a vehicle for more than 2 weeks (14 days).

The price for a private car is R250,00 per day and for a mini-bus R450,00 per day. Clients who hire a mini-bus will receive 15% discount on the rental price for the mini-bus.

The program will receive as input the car type the client wants to rent (private car or mini-bus) and the number of days he/she wants to rent the vehicle for. The client also needs to supply their name, surname and contact number.

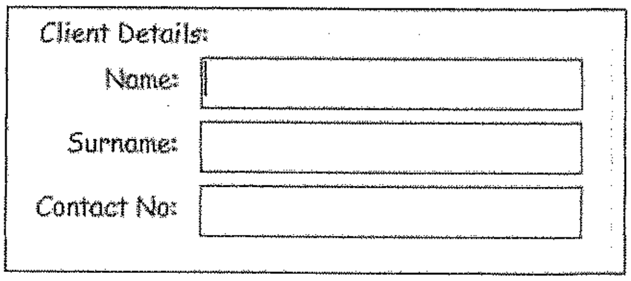
Below is the IPO-chart for the problem:

INPUT	PROCESSING	OUTPUT

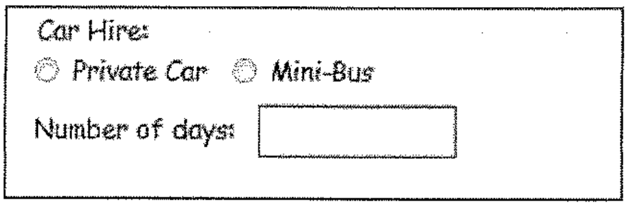
name
surname
contactNo
privateCar / miniBus
noDays
	Car type:
	privateCar
	amountPayable = noDays * 250
	Or
	miniBus
	amtDue = noDays * 450
	discount = amtDue * 0,15
	amountPayable = amtDue – Discount
		amountPayable

The following STEPS explain how to use the program to calculate the *Amount Payable* for the car rental:

STEP 1 RUN the program and enter the client details:



STEP 2 Select the car type and enter the number of days the client will need the car:



STEP 3 Click on the *Compute* button to calculate and display the *amount payable* for hiring of the car:

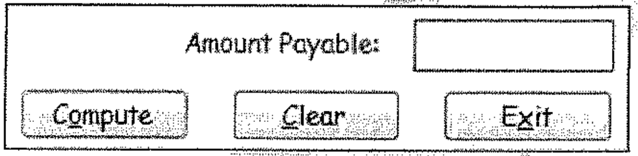


FIGURE 1 below is a print screen of what the user interface looks like. The completed interface has been saved on the hard disk.

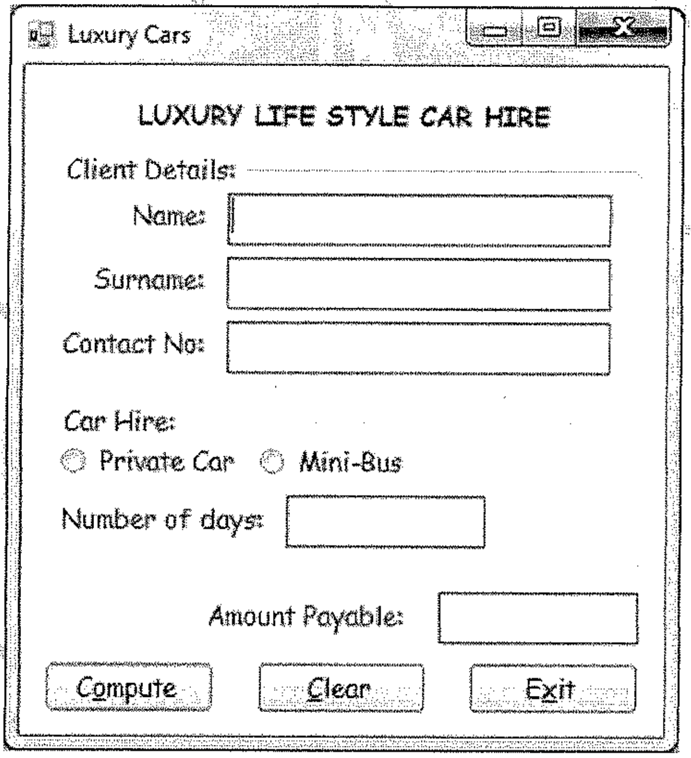
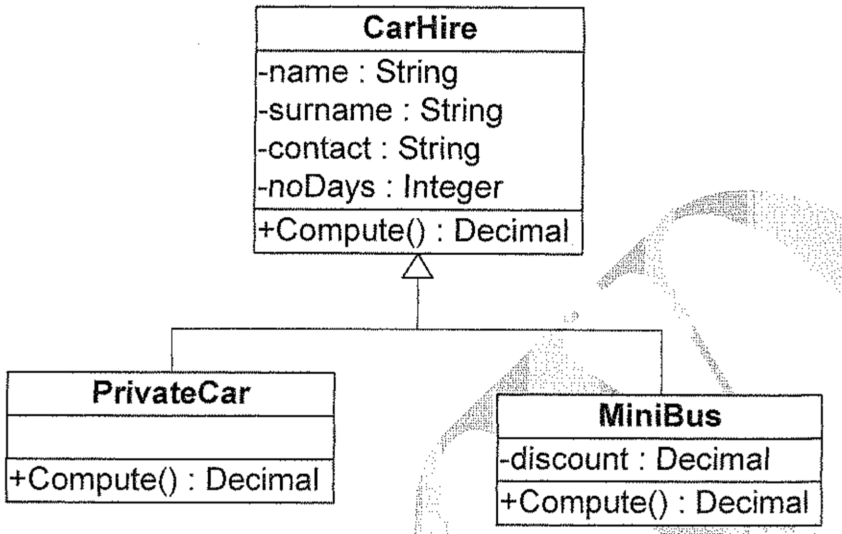


FIGURE 1

Below are the class diagrams of the base/parent class (*CarHire*) and the two derived/child classes (*PrivateCar* and *MiniBus*). Study the class diagrams and answer questions (1.1–1.5).



Open the solution *LuxuryCar* and follow the instructions to complete the program. The base class *CarHire* has been added to the project.

1.1 Add your EXAMINATION NUMBER to the program code of the *CarHire* class. Then add the following program code to *this* class:

1.1.1 Declare the private variables for the *CarHire* class. (1)

1.1.2 Create ALL the public properties for the *CarHire* class. The *noDays* may not be more than 14 days. Add error-trapping code that will throw an exception from within the property procedure if the *noDays* exceeds 14 days. (5)

1.1.3 Create the *Compute()* method for the *CarHire* class. (1)

1.2 1.2.1 Create the *PrivateCar* derived class. Add ALL the fields, properties and methods to the class. (3)

1.2.2 Create the *MiniBus* derived class. Add ALL the fields, properties and methods to the class. The *discount property* of the *MiniBus* class should be read only. (6)

Make a printout of the source code of all THREE classes.

1.3 Add your examination number to the main form of the program. Then add the following program code to this main form:

1.3.1 Write program code to create a *PrivateCar* object. (1)

1.3.2 Write program code to create a *MiniBus* object. (1)

NOTE: The instantiation of the objects should be done in the *Click Event* of the *Compute* button.

1.3.3 Add the following program code to the *Click Event* of the *Compute* button*:*

(a) If the *PrivateCar* radio button has been selected, assign the values from the text boxes to the appropriate properties of the *PrivateCar* object. Invoke the *Compute()* method of the *PrivateCar* class to calculate the amount payable for the private car and display the answer in currency in the appropriate label.

(b) If the *MiniBus* radio button has been selected, assign the values from the text boxes to the appropriate properties of the *MiniBus* object. Invoke the *Compute()* method of the *MiniBus* class to calculate the amount payable for the mini-bus and display the answer in currency in the appropriate label.

(c) Your code should receive an exception from within the *CarHire* class if the *noDays* exceeds 14 days. Handle the exception from within the code of the *Click Event* of the *Compute* button

(6)

Make a printout of the source code.

1.4 Change the text in the title bar of the *Form* to your examination number.

Run the program, enter the following test data and print the form:
Car Type: *PrivateCar*
No of Days: *4*

(1)
[25]

QUESTION 2

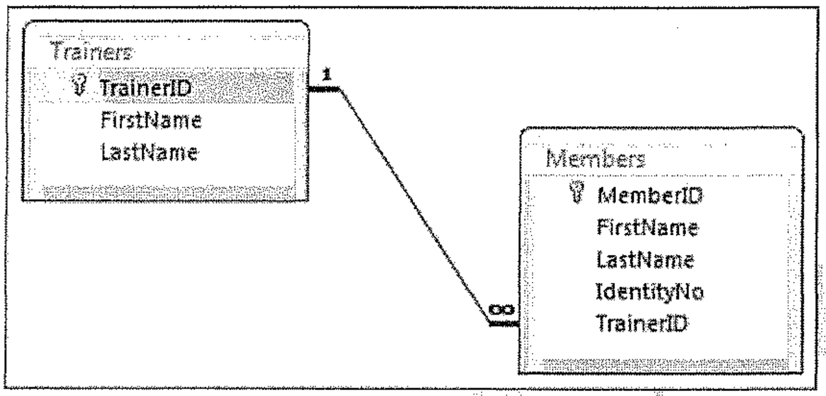
The following questions refer to the VB.NET solution called *UniqueFitness (C:\CP L4 NOV 2013\ UniqueFitness)* that has been saved on the hard drive.

The Unique Fitness Center requested you to design a Visual Basic application for them which they can use to track trainer and member details with. Each trainer may have many members assigned to them, but a member may be assigned to only one trainer.

The VB.NET User Interface will connect to a MS Access database which will allow the user to add and link new members to a specific trainer. The database has already been created and is saved on the hard drive in the application folder:
...\UniqueFitness \UnigueFitness\Bin\Debug\FitnessSystems.mdb

A copy of the database file is saved in the folder:
(C:\CP L4 NOV 2013\ Database\ FitnessSystems.mdb

Below is a graphical representation of the two tables (*trainers* and *members*) and the relationship that exists between the two tables in the database *FitnessSystems.mdb.*



The two TABLES have been created as follows and have been populated with data:

TABLE 1: Trainers

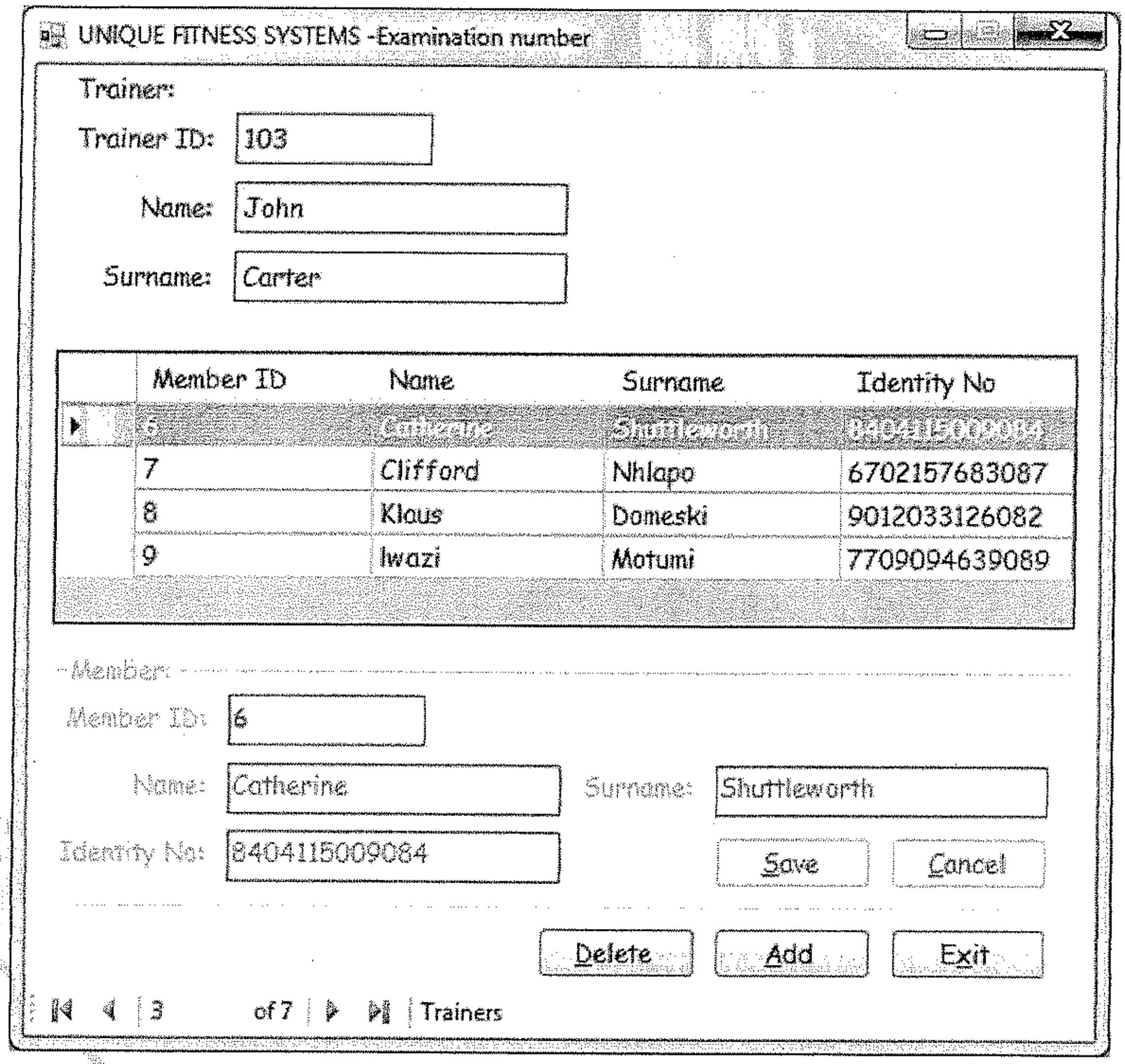
Field Name	Data Type	Field Size
TrainerID	Text	3
FirstName	Text	15
LastName	Text	25

TABLE 2: Members

Field Name	Data Type	Field Size
MemberID	Auto number	4 bytes
FirstName	Text	15
LastName	Text	25
IdentityNo	Text	13
TrainerID	Text	3

The following STEPS explain how to use the program to add records to the *members* table in the *FitnessSystems* database:

STEP 1 RUN the program. A user can scroll to the desired record (*trainer*) by using the navigator at the bottom of the form. The *members* assigned to that specific *trainer* will be listed in the DataGridView.



Trainer:

Trainer ID: 103

Name: John

Surname: Carter

Member ID	Name	Surname	Identity No
6	Catherine	Shuttleworth	8404115009084
7	Clifford	Nhlapo	6702157683087
8	Klaus	Domeski	9012033126082
9	Iwazi	Motumi	7709094639089

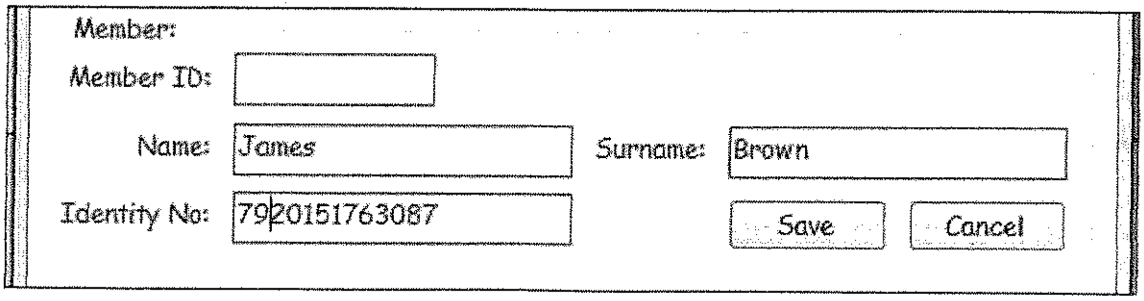
Member:

Member ID: 6

Name: Catherine Surname: Shuttleworth

Identity No: 8404115009084

STEP 2 Once the desired *trainer* has been located click on the *Add* button to add a new *member* to the *members* table. Add the details of the new *member* and click on the *Save* button to save the record to the *members* table. The *MemberID* will be generated automatically.



Member:

Member ID:

Name: James Surname: Brown

Identity No: 7920151763087

NOTE: You are required to use *program code* to add records to the *members* table in the database. NO marks will be allocated if a *data source* has been used to add records.

Open the solution *UniqueFitness.* Add your EXAMINATION NUMBER to the program code and follow the instructions to complete the program:

2.1 Declare a variable to hold the *ConnectionString* of the *connection* object. Assign the provider and data source to the variable as follows:

```
Private connectionPath As String =
'Provider=Microsoft.Jet.OLEDB.4.0;
Data Source=' & Application.StartupPath & '\FitnessSystems.mdb'
```
(2)

2.2 Add the following program code to the click event of the *Save* buttons:

2.2.1 Add a *Try* ... *Catch* block to the program code to ensure that any exceptions that might occur will be trapped and display the error message in a message box. (3)

2.2.2 Declare the *connection* and *command* objects. (2)

2.2.3 Initialise the *ConnectionString* property of the *connection*. (1)

2.2.4 Open the database *connection*. (1)

2.2.5 Set the *Connection* property of the *command* object. (1)

2.2.6 Set the *CommandText* property of the *command* object. (5)

2.2.7 Fetch the values for the *parameters* and add it to the *command* object. (5)

2.2.8 Perform the insertion transaction to the database file. (1)

2.2.9 Close the *connection* to the database. (1)

2.2.10 Call the *AdjustSettings* sub procedure and send a *True* value to the *Usability* parameter. (1)

2.2.11 After you have added a record to the members table you need to refresh the table adapter. Add the following instruction to the program code:

```
Me.MembersTableAdapter.Fill(Me.FitnessSystemsDataSet.Members)
```
(1)

2.3 Change the text in the title bar of the form to your EXAMINATION NUMBER.

Run the program, scroll to *trainer* Peter Parker, add the following *member* and print the form: David Vermeulen, 6703015343085 (1)

Make a printout of the source code.

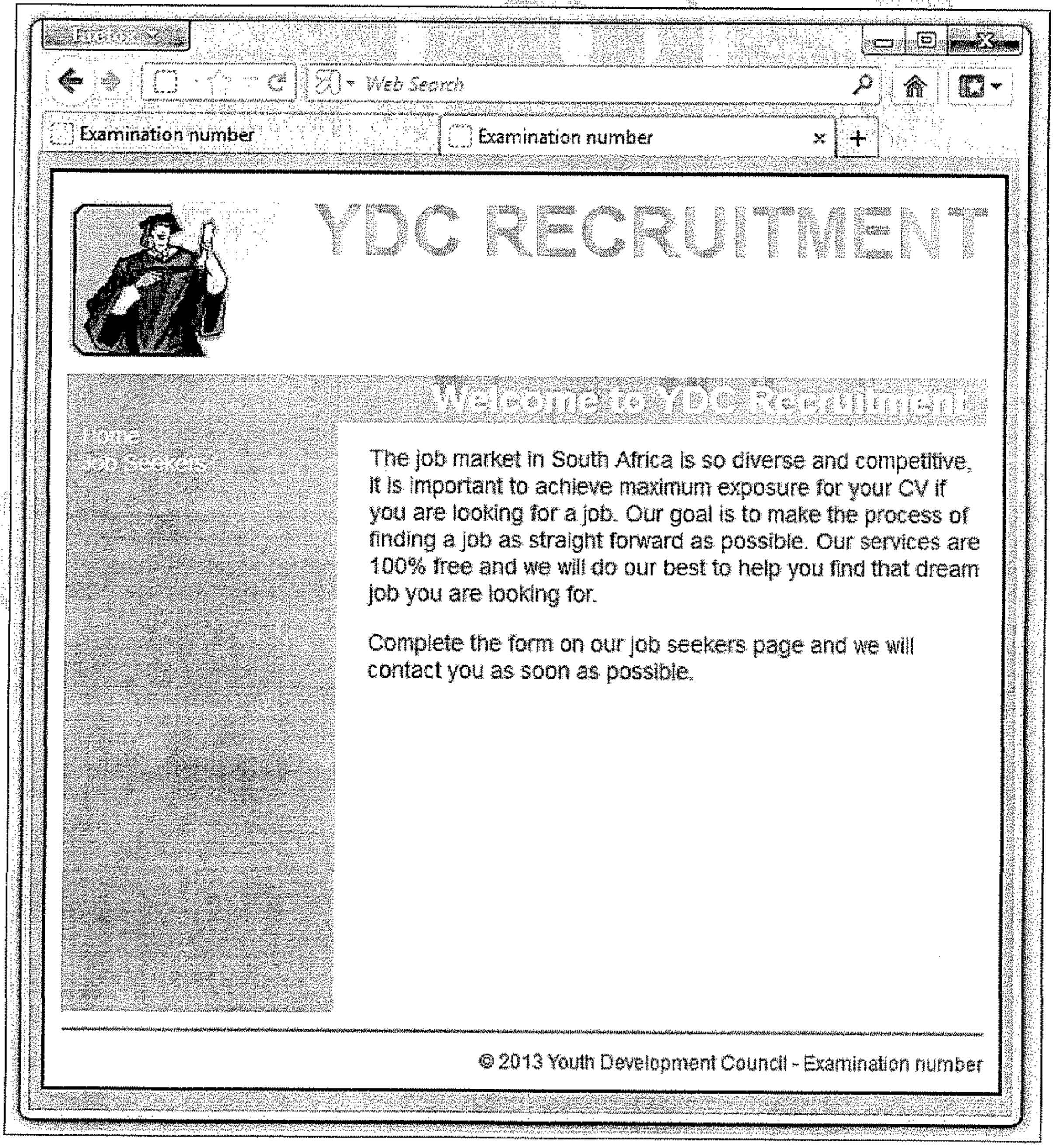
[25]

QUESTION 3

You work for the Youth Development Council as a web developer. You have been tasked to create a web site for the council. The web site consists of 2 web pages: A home page (*Index.html*), and a job seekers page (*JobSeeker.html*). The purpose of the web site is to give new graduates the best possible opportunity to find and apply for their dream job. You have been given clear instructions on how to create each web page.

NOTE: You will design and create the home page (*Index.html*) in question 3 and the job seekers page (*JobSeeker.html*) in QUESTION 4.

Below is a screen print of the home page (*Index.html*):



The job market in South Africa is so diverse and competitive, it is important to achieve maximum exposure for your CV if you are looking for a job. Our goal is to make the process of finding a job as straight forward as possible. Our services are 100% free and we will do our best to help you find that dream job you are looking for.

Complete the form on our job seekers page and we will contact you as soon as possible.

© 2013 Youth Development Council - Examination number

Index.html

The page is divided into SIX divisions:
A *page* division which contains all the other divisions. A *banner* at the top of the page, followed by a *heading* and footer at the bottom of the page. The main part of the web page is divided into two divisions: a *sidebar* on the left and a *main* division on the right hand side of the page.

Copy the following files from the folder: *C:\CP L4 NOV 2013\YouthDevelopment* to your working directory: *Logo.jpg; Content.txt; Styles.css.*

Save all the files in the same directory (HTML, CSS and graphic files).

The text for the main paragraph is typed and saved in the file *Content.txt.*

3.1 Open the external Cascading Style Sheet (CSS) *Styles.css.*

Add your EXAMINATION NUMBER as a comment to the CSS code.

3.1.1 By default, all links in a web page are underlined. Add a rule set which will remove the underline from all links. (2)

3.1.2 Add a rule set for the *img* element in the *banner* division. Set the top margin to .5 ems. Set the right margin to 12 pixels. Float the image to the left. (3)

3.1.3 Add a rule set for the *sidebar* division. Set the width to 150 pixels and the height to 370 pixels. Set the background colour to #9999CC. Add padding to the left and right of 10 pixels and set the bottom margin to .7 ems. Float the division to the left. (6)

3.1.4 Create a rule set for a *copyright* class. Set the margin to 0, align the text to the right and set the size of the font to 90%. (3)

Make a printout of the CSS code.

3.2 Create an HTML file and save it as *Index.html.* Insert your EXAMINATION NUMBER as the title of the page.

3.2.1 Add a link to the external style sheet *Styles.css.* (2)

3.2.2 Define the body section with SIX div elements. The first element is the *page* division and it will contain everything in the body section. Within this div element code FIVE more div elements: banner, header, sidebar, main and footer. (1)

3.2.3 Add the *Logo.jpg* to the *banner* division. Add an h1 element with the text 'YDC RECRUITMENT' to this division. (1)

3.2.4 Add an h1 element with the text 'Welcome to YDC Recruitment' to the *header* division. (1)

3.2.5 Add links to the home page and the job seekers page to the *sidebar* division as follows:
'Home' (*Index.html*)
'Job Seekers' (*JobSeeker.html*) (1)

3.2.6 Add the paragraph 'The job market in South Africa ... ' to the *main* division. (1)

3.2.7 Add the paragraph 'Complete the form on our job seekers page and we will contact you as soon as possible.' to the *main* division. The text 'job seekers' must link to the job seekers page. (2)

3.2.8 Add a paragraph to the *footer* division that includes the copyright symbol and the information shown on the web page with your EXAMINATION NUMBER as follows:

'© 2013 Youth Development Council - Examination number'

Use the *copyright* class to display the text. (2)

Make a printout of the HTML code.

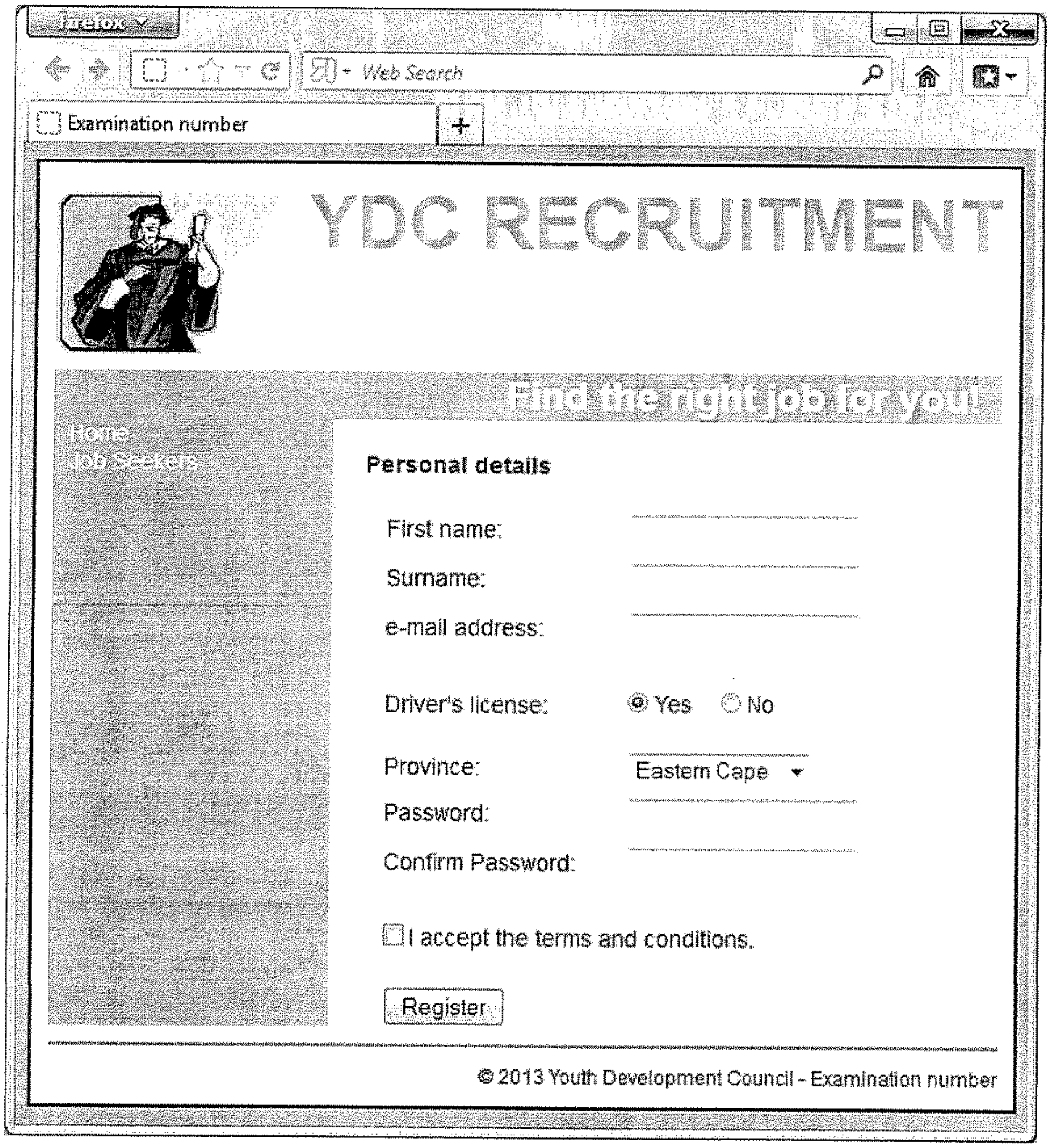
Launch the page from your web browser and print the page directly from the web browser.

[25]

QUESTION 4

Create the job seekers page for the web site that you created in QUESTION 3.

Below is a screen print of the web page (*JobSeeker.html*):



Personal details

First name:

Surname:

e-mail address:

Driver's license: [✓ Yes] No

Province: Eastern Cape

Password:

Confirm Password:

☐ I accept the terms and conditions.

Register

© 2013 Youth Development Council - Examination number

JobSeeker.html

HINT: The job seeker page looks exactly the same as the home page (*Index.html*). Save a copy of the *Index.html* page and name it *JobSeeker*.html.

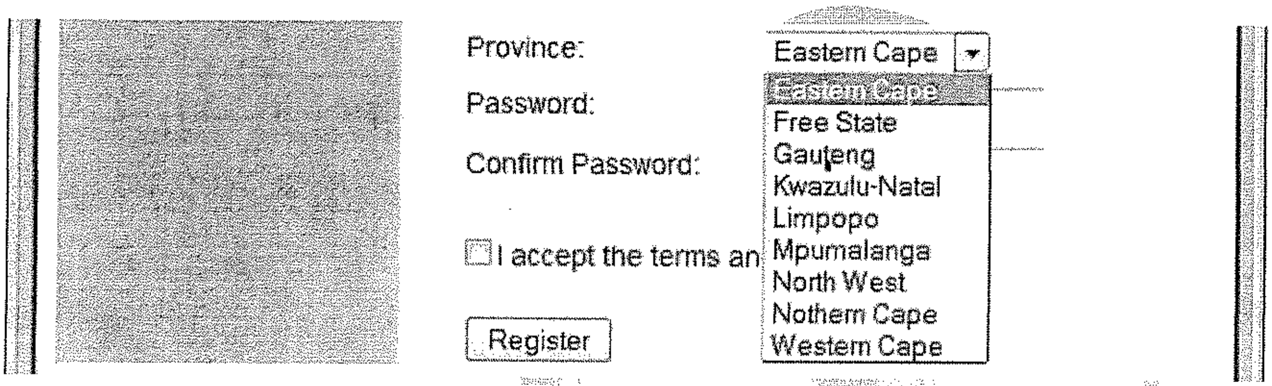
4.1 4.1.1 Create a form element inside the *main* division of the web page and name it *info*. (1)

4.1.2 Add an h4 element with the text 'Personal details'. (1)

4.1.3 Add the text fields for *First name*, *Surname* and the *e-mail address* to the form. (1)

4.1.4 Add the radio buttons to the form and check the 'Yes' button. (3)

4.1.5 Add the following drop-down list to the form and select 'Eastern Cape' from the list: (4)



Province: Eastern Cape

Password:

Confirm Password:

I accept the terms an

Register

Eastern Cape
Free State
Gauteng
Kwazulu-Natal
Limpopo
Mpumalanga
North West
Nothern Cape
Western Cape

4.1.6 Add two password fields to the form and name it *pwd* and *confirm_pwd*. Set the maximum length of the password fields to 6 characters. The *confirm_pwd* field allows the user to retype his password to ensure that the user entered the intended password. (2)

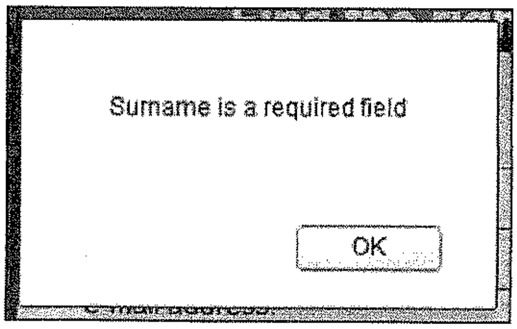
4.1.7 Add the check box to the form. The text 'terms and conditions' must link to a page *TermsConditions.html* (The page does not actually exist). (2)

4.1.8 Add the *Register* button which will call a *JavaScript* function called *Register()* when the button is clicked. (3)

4.2 4.2.1 Add a *JavaScript* element to the web page. (1)

4.2.2 Create a *JavaScript* function called *Register()* to validate the input on the web page and create variables to hold the values from the *surname*, *password* and *confirm password* fields. (1)

4.2.3 The *surname* field may not be empty. If empty, display an error message in an alert box, terminate the function and return focus to the *surname* text field.

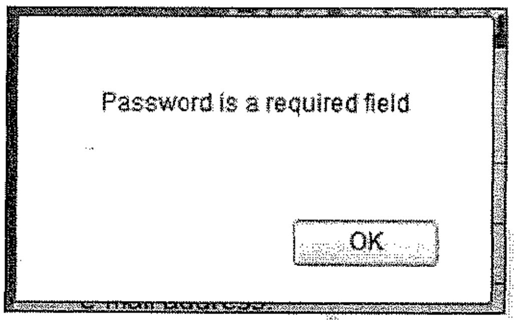


Surname is a required field

OK

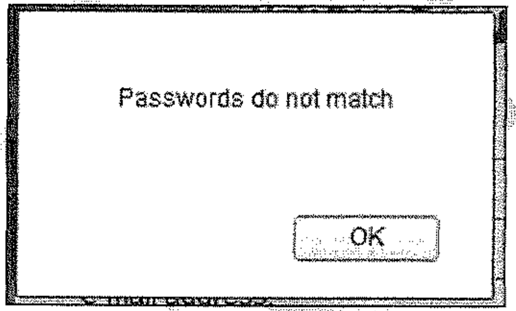
(2)

4.2.4 The *password* field may not be empty. If empty, display an error message in an alert box, terminate the function and return focus to the *password* field.


Password is a required field

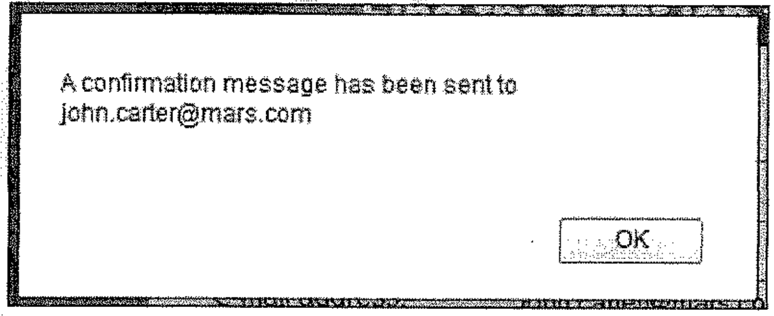
(1)

4.2.5 The values from the *password* and *confirm password* fields must match. If they do not match, display an error message in an alert box, terminate the function and return focus to the *password* field. Clear the values from the *password* fields.


Passwords do not match

(1)

4.2.6 Display the following message if all the entries on the web page is valid.


A confirmation message has been sent to john.carter@mars.com

(1)

Ensure that your examination number appears in the code of the web page. Make a printout of the source code.

4.2.7 Launch the web page from your browser and enter the following test data:
Bruce Wayne, bruce.wayne@arkham.com

Password: catwoman

Make a printout of the alert box. (1)

[25]

TOTAL: 100