Tutorial letter 104/2/2015

Introduction to Programming II

COS1512

Semester 2

School of Computing

Examination Tutorial Letter

Bar code

Learn without limits.

UNISA university of south africa

CONTENTS

1 PURPOSE OF THIS LETTER

Dear Student,

This tutorial letter contains the information pertaining to the examination as well as an examination memorandum for the May 2014 examination paper. Please make sure that you keep this letter at hand and read it often while preparing for the examination.

2 TUTORIAL MATTERS UPLOADED

All the study material except the prescribed text book has been uploaded on our web learning portal (myUnisa). Please make sure that you visit myUnisa in order to download the material that has been uploaded. Since this module is offered mostly online, you will only receive Tutorial Letter 101 by post. Please click the following link to access myUnisa (https://my.unisa.ac.za.) and make sure that you download all the tutorial letters available on myUnisa.

3 EXAMINATION ADMISSION

The university will inform you whether you obtained examination admission or not. It will also provide you with details of the examination (date, time and the venue). It is your responsibility to make sure that the date, time and examination centre are correct. If you need any further information or details, please contact UNISA's Examination department. You can contact the Examination department on the following contact details:

College of Agriculture & Environmental Sciences AND College of Science, Engineering and Technology			
M Vosloo	General examination enquiries	aegrotats@unisa.ac.za exams@unisa.ac.za examdisabled@unisa.ac.za examadmission@unisa.ac.za examinternational@unisa.ac.za remark@unisa.ac.za purchasescript@unisa.ac.za	011 471 2928
B Molefe			011 471 3121
Examination Venue and Invigilation Administration			
IT Zulu	General student enquiries in respect of examination venues and examination invigilators.	invigilationandexamvenues@unisa.ac.za	011 471 3174
ZB Makhabane			011 471 2375

Examination admission will be based on whether or not you have submitted your assignment before the cut-off date which has been specified. If you have problems with examination admission, please make sure that you contact the Examination department. Please note that lecturers are not responsible for examination queries.

4 EXAMINATION

Please note the following:

- The exam paper covers all the work which has been outlined in tutorial letter 101 and specifically tests the outcomes specified in the tutorial letter 101. You need to use the prescribed book (Savitch 7th, 8th or 9th edition) as well as all COS1512 tutorial letters for your examination preparation.
- Work through all your assignments and tutorial letters providing solutions to the assignments. **We included additional notes in the solutions that should assist you in understanding the work.**
- Do not ignore assignment 3. Assignment 3 is self-assessment. Students frequently fail the COS1512 exam because they have not mastered the last part of the study material, which is covered by assignment 3.
- The examination consists of a single **2-hour** paper.
- The total for the examination paper is 75 marks.
- The examination is a closed book examination, i.e. you are not allowed to bring any notes or material to the examination centre.
- No calculators are allowed during the examination.
- There will be no multiple choice questions.
- The examination will be written on paper in an answer book.
- NOTE: When you work through the old exam papers, write out the answers completely. You have to practice to write it down, as you don't have a compiler in the exam to point out compiler errors.

- **Note that some questions may cover more than one learning outcome or topic.**

During the examination, ensure that you:

- Read the exam paper calmly and properly before you start answering the questions
- Read the instructions and follow them.
- Read the answers again after completing the examination.
- Complete the questions that you struggled with.

It is advisable to read the exam paper before you start answering questions. Make sure that you have completed all the answers before submitting your answer book or examination script.

5 LAYOUT OF EXAMINATION PAPER

This section provides estimates of the marks allocated per question to different sections of the work. You will need to know all the work which has been outlined in Tutorial Letter 101. The mark allocation will approximately be as follows:

Question 1: +/- 4 marks: C strings
Question 2: +/- 4 marks: recursion
Question 3: +/- 5 marks: pointers
Question 4: +/- 27 marks: class definition and implementation
Question 5: +/- 10 marks: file processing
Question 6: +/- 15 marks: inheritance
Question 6: +/- 10 marks: vectors and templates

Please note that we test all the work. This outline points out the most important aspects tested in each question.

6 PREVIOUS EXAMINATION PAPER

The examination paper which is used for this section is on myUnisa. It is the May/June 2014 examination paper. Please download the examination paper. We have only included the memorandum for the examination.

Memorandum for May 2014 Exam

QUESTION 1 **[8]**

1.1 (6)

```
double computeBill(double price) √
{
          return price * 1.14;  √
}
double computeBill(double price, int quantity)    √
{
     return (price * quantity) * 1.14;          √
}
double computeBill(double price, int quantity, double coupon)     √
{
     double total = (price * quantity - coupon) * 1.14;               √
     return total;
}
```

1.2 (2)

```
int main()
{
     double price = 12.99;
     int quantity = 10;
     double coupon = 5.00;
```

```
    cout << "Amount = " << computeBill(9.99);
    cout << "Amount = " << computeBill(9.99, 10);          √√
    cout << "Amount = " << computeBill(9.99, 10, 50.00);
}
```

QUESTION 2 **[6]**

2.1 (3)

```
Rectangle duplicate (Rectangle & rect1 ½, Rectangle & rect2½)
{
    rect1.width ½ = rect2.width; ½
    rect1.height ½ = rect2.height; ½
}
```

2.2 (3)

```
friend ½ Rectangle ½ duplicate (Rectangle & rect1, Rectangle & rect2); √

duplicate (foo, bar); √
```

QUESTION 3 **[5]**

3.1 `SS.push_back("The number is 10");` √ (1)

3.2 `cout << SS.size() << endl;` √ (1)

3.3 `cout << SS[2] << endl;` √ (1)

3.4 (2)

```
int ii;    ½
for(ii=0 ½; ii < SS.size()½; ii++)
{
   cout << SS[ii] ½ << endl;
}
```

QUESTION 4 **[2]**

4.1 `if (n == 0)` √ (1)

4.2 `We are busy with recursion111` √ (1)

QUESTION 5 **[5]**

1. `#include <fstream>  // file i/o` √

2. `ifstream input;` √

3.
```
input.open(name);          ½
if(!input)                 ½
{
   cout << "Cannot open file " << name << " Aborting." << endl;
```

```
    exit (1);
}
```

4. `while(input)` √

5. `input.close();` √

QUESTION 6 **[10]**

6.1.1 The declaration `int * pOne;` declares `pOne` to be a pointer pointing to an `int` value. √ (1)

6.1.2 The declaration `int vTwo;` declares `pTwo` as an `int` value. √ (1)

6.1.3 The statement `int * pThree = &vTwo;` declares a pointer to an integer and initializes it with the address of another variable of `vTwo`. √ (1)

6.2.1 The declaration `int * q = p;` declares `q` to be a pointer pointing to the same `int` to which `p` points. √
The assignment `n = *p;` assigns to `n` the `int` to which `p` points. √ (2)

6.2.2 The declaration `int & r = n;` declares `r` to be a reference for the `int` variable `n`. √ (2)
The assignment `p = &n;` assigns the address of `n` to the pointer `p`. √

6.3.1 1 2 3 4 5 6 7 8 9 √

6.3.2 10 1 2 3 4 5 6 7 8 9 √

6.3.3 0 1 2 3 4 5 6 7 8 9 √

QUESTION 7 **[23]**

7.1 (8)

```
class Product
{
      public:
            Product();
            Product(long, double, double, long); √
            void display();       √
            double retailPrice(); √
            void modify();   √
            Product increment();  √
            Product decrement();  √
       private:
            long id;         ½
            double price;    ½
            double markup;   ½
            long number;     ½
};
```

7.2 (10)

```
#include <iostream>
```

```
#include <iomanip>
#include <string>
#include "product.h"  √
using namespace std;

Product::Product()
{
      id = 0;
      price = 0.0;          √
      markup = 0.0;
      number = 0;
}

Product::Product(long new_id,double new_price,double new_markup,long
      new_no)
{
      id = new_id;          ½
      price = new_price;    ½
      markup = new_markup;  ½
      number = new_no;      ½
}

void Product::display()          √
{
      cout << "The product information is as follows:" << endl;
      cout << setiosflags (ios::left) << setw(10) << "Product Id"
                  << setw(10) << "Price" << setw(10) << "Markup %"
                  << setw(10) << "Number" << endl;
      cout << setiosflags (ios::left) << setw(10)<< id << setw(10)
                  << price << setw(10) << markup << setw(10)
                  << number << endl;
}
double Product::retailPrice()
{
      return ((markup / 100) * price); √
}
void Product::modify()
{
      double new_markup;                ½
      cout << "Please enter new markup percentage: ";
      cin >>  new_markup;               ½
      while ( (new_markup < 0 ) || (new_markup > 100) )
      {
            cout << "Incorrect markup percentage - Please renenter : ";
            cin >>  new_markup;
      }
      markup = new_markup;              √

}
```

```
Product Product::increment()
{
      number++;        √
      return *this;
}

Product Product::decrement()
{
      number--;        √
      return *this;
}
```

7.3 (2)

Program increment the current instance by 1 and then decrement it twice. √

```
      The product information is as follows: √
      Product IdPrice     Markup %  Number
      1111      10        20        99
      Retail Price   :2
```

7.4. (3)

```
Product Product::operator++ (int)  √
{
      Product x = *this;     √
      number++;
      return x;              √

}
```

.

QUESTION 8 **[8]**

8.1 (4)

```
template <class T>    √
class ComplexNumber
{
    public:
      ComplexNumber();
      ComplexNumber(T realpart);
      ComplexNumber(T r, T i);        √
      void output(char&i);
      ComplexNumber<T> add(ComplexNumber<T> x);  √

    private:
        T real;       ½
        T imagine;    ½
};
```

8.2 (3)

```
template <class T>    √
```

```
ComplexNumber<T>::ComplexNumber(T r, T i) √
{
    real = r;    ½
    imagine = i; ½
}
```

8.3 (1)

```
ComplexNumber<double> c1(3,6);   √
```

QUESTION 9 **[8]**

For each of the following Questions create the header files

9.1 (3)

```
class door
{
    public:
        door();              ½
        int isOpen();        ½
        void Open();         ½
        void Close();        ½
    protected:
        state  door_status;  √
};
```

9.2 (3)

```
#include "door.h"                √
enum lock_status {locked, unlocked};

class lockdoor : public door     √

{
    public:
        lockdoor();          //constructor
        int isLocked();
        void Open();
        void Lock();
        void unLock();
    protected:
        lock_status      lockdoor_status;     √

};
```

9.3 (2)

```
door       aDoor;          √

if (aDoor.isOpen() )       ½
```

```
        aDoor.Close();          ½
else
        aDoor.Open();
```