UNIVERSITY EXAMINATIONS

UNIVERSITEITSEKSAMENS

UNISA university of south africa

COS1512
RCO1512

October/November 2016

INTRODUCTION TO PROGRAMMING II

Duration 2 Hours

75 Marks

EXAMINERS

FIRST	MRS HW DU PLESSIS	MRS MA SCHOEMAN
SECOND	MS A THOMAS	

Closed book examination

This paper consists of 7 pages and 6 questions.
Please make sure that you have all 7 pages with the 6 questions

Instructions / *Instruksies*:

- Answer all the questions
- Do all rough work in the answer book
- The mark for each question is given in brackets next to the question.
- Please answer the questions in the correct order If you want to do a question later, leave enough space
- Number your answers and label your rough work clearly
- Marks are awarded for part of an answer, so do whatever you are able to in each question

GOOD LUCK!

QUESTION 1 [5]

A palindrome is a word that reads the same from the front as from the back, e g `kayak` The following code fragment reverses the letters of a word, and then tests if it is equal to the original word Consider the code and then answer the questions that follow

```
1     char temp, word[6] = "kayak", saveWord[6] = "games",
2     int j  = 5,
3     saveWord = word;
4     for (int i = 0; i < (j+1)/2, i ++)
5     {
6           temp = word[i],
7           word[i] = word[j],
8           word[j] = temp,
9           j--,
10    }
11    if (word != saveWord)
12       cout << "Word is not a palindrome  " << endl,
13    else cout << "Word is a palindrome  " << endl;
```

1 1 Explain why the instruction in line 3 is problematic Give the correct code to assign the value of `word` to `saveWord` (2)

1 2 Explain why the instruction in line 11 is problematic Give the correct code to test if the value of `word` is **not** the same as the value of `saveWord` (1)

1 3 The output of this code is not what we expect It actually outputs that `kayak` is not a palindrome Why? How would you change the code in line 2 to fix the problem (2)

QUESTION 2 [5]

Consider the following program and then answer the questions that follow

```
1  #include <iostream>
2  using namespace std;
3  void count(int counter)
4  {
5
6  if (counter == 0)
7    return,
8  else
9  {
10  cout << "Calling counter(" <<counter << ")" << endl;
11  count(--counter),
```

```
12  cout << "counter(" << counter << ") processed  " << endl,
13  return;
14 }
15 }
16  int main()
17  {
18  int i = 3,
19  count(i);
20  return 0;
21  }
```

2 1 What is the output written to the standard console when `main()` is executed? (2)

2 2 Identify the base case in the recursive function `count()` (1)

2 3 Identify the general case in the recursive function `count()` (1)

2 4 Write down an instruction for line 5, using the `assert()` function to ensure that `counter` is not negative Write down only the instruction – do not copy the program in your answer book (1)

QUESTION 3 [7]

Consider the following code fragment which is assumed to be embedded in a complete and correct C++ program

```
1     int *p1 = new int, *p3,
2     *p1 = 25,
3     int a, b, *p2 = &b,
4     *p2 = *p1 - 10,
5
6     *p3 = &a,
7     b =*p3 + a,
8     delete p1,
9     p2 = p1,
10    cout << a + *p3 << ' ' << *p2 << ' ' << b << endl,
```

3 1 Give an assignment statement for line 5 that assigns twice the value of the variable that `p1` is pointing to, divided by 5, to `a` (2)

3 2 Explain the purpose of the statement `int *p1 = new int,` in line 1 (1)

3 3 Why is line 9 a potentially risky statement? (1)

3 4 In line 6 we want to assign the address of variable `a` to `p3` This statement is incorrect. Give the correct statement (1)

3 5 Explain the purpose of * in line 7 (1)

3 6 Assuming that you have given a statement for line 5, corrected line 6, and that line 8 is not executed at all, what is the output after line 10 has been executed? (1)

QUESTION 4 [31]

Use separate compilation to define a class called `Bursary` that represents a student study record This class should have three member variables

- `studentNumber`, an integer variable that holds the student number
- `yearsOfStudy`, an integer variable that holds the study year of the student, e g a value of `2` represents the student is in his/her second year of study
- `modulesPassed`, an integer that holds the number of modules already passed by the student

In addition, the class should contain the following member functions

- A default constructor that initializes `studentNumber`, `yearOfStudy` and `modulesPassed` to `0`
- An overloaded constructor that accepts a student's details and sets `studentNumber`, `yearOfStudy` and `modulesPassed` to specified values
- A destructor that does not perform any action
- An accessor function `get_number()` to return the value stored in an object's `studentNumber` member variable.
- A mutator to update the member variable `yearsOfStudy` to a specific value
- An overloaded equality operator`==` to compare a student's study year and number of modules passed, to the criteria required to be taken into account for a bursary The `==` operator is implemented as a **`friend`** function with the following prototype

```
friend bool operator==(const Bursary & bursary1,const Bursary & bursary2),
```

This function returns `true` if `bursary1` has the same value for `yearOfStudy` and `modulesPassed` as `bursary2`

- An overloaded extraction operator `>>` (implemented as a **`friend`** function) so that it can be used to input values of type `Bursary` from any file

You should attempt the solutions as follows

4 1 Create the header file `bursary h` that contains the `Bursary` class specification (9)

4 2 Create the implementation of the class `Bursary` including all the **`friend`** functions (11)

4 3 Demonstrate the class in an application program (`main()`) that is used to list all the students that qualify for a bursary Allow the user to enter the required year of study and number of modules passed to qualify for a bursary Use the overloaded constructor to initialise a `Bursary` object named `criteria` to the year of study and number of modules the user specified (initialize the student number for this object to `0`)

Assume that the students' study records are stored in a file `Bursary dat` Create a `Bursary` object for each student record, use a `while` loop to input the student record from `Bursary.dat`, use the overloaded equality operator== to compare the year of study and number of modules passed in the student record read from `Bursary dat` one by one with `criteria`, and display a list of all the student numbers for students that have the same year of study and number of modules passed as the specified criteria (10)

4 4 Why are the overloaded operators == and >> implemented as **`friend`** functions? (1)

QUESTION 5 [15]

The class `CellContract` below describes a cell phone contract Consider the specification (interface) of this class:

```
class CellContract
{
public
     CellContract(),
     CellContract(int minutes, int data),
     void displayBalances()const,
     void getBalances(int &minutes, int &data)const,
     void getContactdetails(string &name, string &address)const,
     void setContactdetails(string name,
                            string contactdetails),
  private
      int talkTime,
      int dataMB,
      string number,
      string name,
      string address,
  },
```

5 1 Derive a class `Topup` from the class `CellContract` This class has an additional member variable, `SMS` The class `Topup` also has member functions, `addAirtime()` that adds minutes to `talkTime`, `addData()` that adds a number of MB (megabytes) to `dataMB` and `addSmsBundle()` that increases `SMS` with the specified number of sms's The class `Topup` should override function `getBalances()` in order to include all the member variables of `Topup`

Provide only the interface of class `Topup` in a header file The header file should contain compiler directives to prevent multiple definitions Assume that the interface of the class `CellContract` is contained in a file called `"CellContract h"` (9)

5 2 Implement the overloaded constructor for the class `Topup` by invoking the base class constructor (3)

5 3 Consider the following implementation

```
void Topup  displayBalances()const
{
      cout << "Talktime minutes available :" << Talktime
          << endl,
      cout << "MB data available :" << MBData << endl,
      cout << "Number of SMSs available  " << SMS << endl;
}
```

Explain why this is not a legal definition in the derived class `Topup`? (1)

5 4 Is the function `displayBalances()` an example of overloading? Explain your answer (2)

QUESTION 6 [12]

The *Foodzone Superstore* keeps record of their cashiers' login IDs, passwords and terminal numbers that they worked on The following class allows the user to add a new cashier's details to the record or enter a cashier's login ID, and then return the cashier's password

```
class CashierList {
      public
            CashierList(),
            void addOne(int pcashier, const string &pword,
                        int pterm);
            string lookup(int pcashier) const;

      private
            vector <int> cashier;
            vector <string> password,
            vector <int> terminal;
}
```

The class `CashierList` has the following operations

`addOne()` – adds a new cashier's details to the record

`lookup()` – returns the password of the cashier ID entered

6.1 Write a template version of the class `CashierList` Use the template prefix `CashierList <TCashier, TPWord, TTerm>` to re-design the `CashierList` interface, so that the cashier ID or terminal number, or the password could be of different data types (5)

6 2 Implement the `addOne()` function of the template class `CashierList` (5)

6 3 Write the code to create an object of the class `CashierList` that has a cashier ID of type `string`, a password of type `double` and a terminal number of type `int` (2)