| hrs | / /20 | | Exams Office Use Only |
|---|---|---|---|

University of the Witwatersrand, Johannesburg

| | |
|---|---|
| Course or topic No(s) | ELEN3009 |
| Course or topic name(s)<br>Paper Number & title | Software Development II |
| Examination/Test* to be held during month(s) of<br>(*delete as applicable) | November 2018 |
| Year of Study<br>(Art & Sciences leave blank) | Third |
| Degrees/Diplomas for which this course is prescribed<br>(BSc (Eng) should indicate which branch) | BSc(Eng)(Elec) |
| Faculty/ies presenting candidates | Engineering and the Built Environment |
| Internal examiners and telephone number(s) | Dr SP Levitt x77209 |
| External examiner(s) | Mr K Ortlepp |
| Special materials required (graph/music/drawing paper) maps, diagrams, tables, computer cards, etc) | Computer card for multiple-choice questions |

Time allowance

| Course Nos | ELEN3009 | Hours | 3 |
|---|---|---|---|

Instructions to candidates
(Examiners may wish to use this space to indicate, inter alia, the contribution made by this examination or test towards the year mark, if appropriate)

a) Read instructions on page 1 of exam
b) Available marks: 110 - Full marks: 100
c) Closed-book exam
d) Basic scientific calculator allowed

Internal Examiners or Heads of School are requested to sign the declaration overleaf

1. As the Internal Examiner/Head of School, I certify that this question paper is in final form, as approved by the External Examiner, and is ready for reproduction.

2. As the Internal Examiner/Head of School, I certify that this question paper is in final form and is ready for reproduction.

(1. is applicable to formal examinations as approved by an external examiner, while 2. is applicable to formal tests not requiring approval by an external examiner—Delete whichever is not applicable)

Name:_____ Signature:

(THIS PAGE NOT FOR REPRODUCTION)

**Instructions**

- Answer *all* questions. The questions do not carry equal weight.
- The paper is divided into two parts:
  - **Part A** consists of several multiple choice questions which are to be answered on the computer card provided. You must fill in your student number on the card. There may be more than one correct answer for each of the multiple choice questions, for example (a) and (e). Indicate ALL the correct answers for each question by clearly marking the appropriate blocks of the chosen letters on the card with a dark HB pencil. A negative marking system will be adopted so **DO NOT GUESS**. Marks will be subtracted for incorrect answers. You cannot, however, get less than zero for any single multiple choice question.
  - **Part B** consists of three (3) questions to be answered legibly in the answer book provided.
- For questions which require you to write **source code**, note that:
  - Pencil may be used.
  - You only need to specify `#include`'s if specifically asked.
  - For classes, you can give the implementation entirely in the header file, unless directed otherwise.
  - Marks are not awarded solely for functionality but also for good design, making appropriate use of library functions, following good coding practices, and using a modern, idiomatic C++ style.
  - Your code must be easily understandable or well commented.
- Reference sheets are provided in a separate appendix.

# Part A

## Question 1

1.1 When writing code, it is considered good practice to: (5 marks)

(a) Rely on comments to explain how a function is implementing its task.

(b) Name types using nouns, and functions using verbs.

(c) Describe the return value type within the function name so that programmers have a better understanding of the function.

(d) Use abbreviations in variable and function names.

(e) Phrase functions that return a Boolean value as questions.

1.2 Given the following program, which of the statements concerning it are true? (5 marks)

```
1  int calculate_length(const string& word) { return word.length(); }
2
3  int main()
4  {
5     auto words = vector<string>{"This", "is", "a", "short", "sentence"};
6     auto word_lengths = vector<int>(words.size()); // for results
7     transform(begin(words), end(words), begin(word_lengths),
          calculate_length);
8     return 0;
9  }
```

(a) An in-place transformation can be used instead of storing the results in `word_lengths`.

(b) The `transform` function does not call the `calculate_length` function when the iterator moving over the range is equal to: `end(words)`.

(c) The following line of code is equivalent to that on line 6:
`auto word_lengths = vector<int>{words.size()};`

(d) `calculate_length` is known as a *function object*.

(e) The implementation of the `transform` function does not use `vector<T>::push_back`.

1.3 Which of the following statements concerning inheritance are true? (5 marks)

(a) The derived class inherits the interface of the base class.

(b) When a protected member of a base class is publicly inherited, it becomes public in the derived class.

(c) Given the following definition of `class B`

```
class B : public A {...}
```

the order of the execution of the class constructors will be `B` followed by `A`.

(d) The member functions of a child class can directly access only the protected and public data members of the parent class.

(e) A class can inherit member functions and data members from more than one class which is known as multiple inheritance.

1.4 Which of the following statements are correct? (5 marks)

(a) `make_shared` and `make_unique` are not susceptible to memory allocation failures.

(b) Only shared pointers may be passed by value.

(c) Reassigning a smart pointer to a new pointee always releases the memory used by the old pointee.

(d) Smart pointers are usually constructed on the stack.

(e) When a `unique_pointer` is copied, its pointee is also copied.

1.5 Which of the following statements are true? (5 marks)

(a) Static member functions cannot be overloaded.

(b) Static member functions can only access static data members and other static member functions.

(c) Static data members can only be accessed by static member functions.

(d) Non-static data members can be accessed by static member functions.

(e) The output of the following program is: `1 2 3`

```
class Player
{
private:
    int id;
    static int next_id;
public:
    int getID() { return id; }
    Player()  {  id = next_id++; }
};

int Player::next_id = 1;

int main()
{
  vector<Player> players(3);
  cout << players[0].getID() << " ";
  cout << players[1].getID() << " ";
  cout << players[2].getID() << " ";
  return 0;
}
```

1.6 Which of the following statements concerning Git and GitHub are true? (5 marks)

(a) Git is a centralised version control system

(b) Assuming that you have a linear commit history, changing any commit in the history will cause the commit hashes of all of the subsequent commits to change.

(c) The `rebase` command allows you to combine commits together.

(d) The `commit` command pushes code contained in a local repository up to a remote repository.

(e) A pull request on GitHub is primarily used for code review.

1.7 Which of the following statements concerning Figure 1 are true? (5 marks)

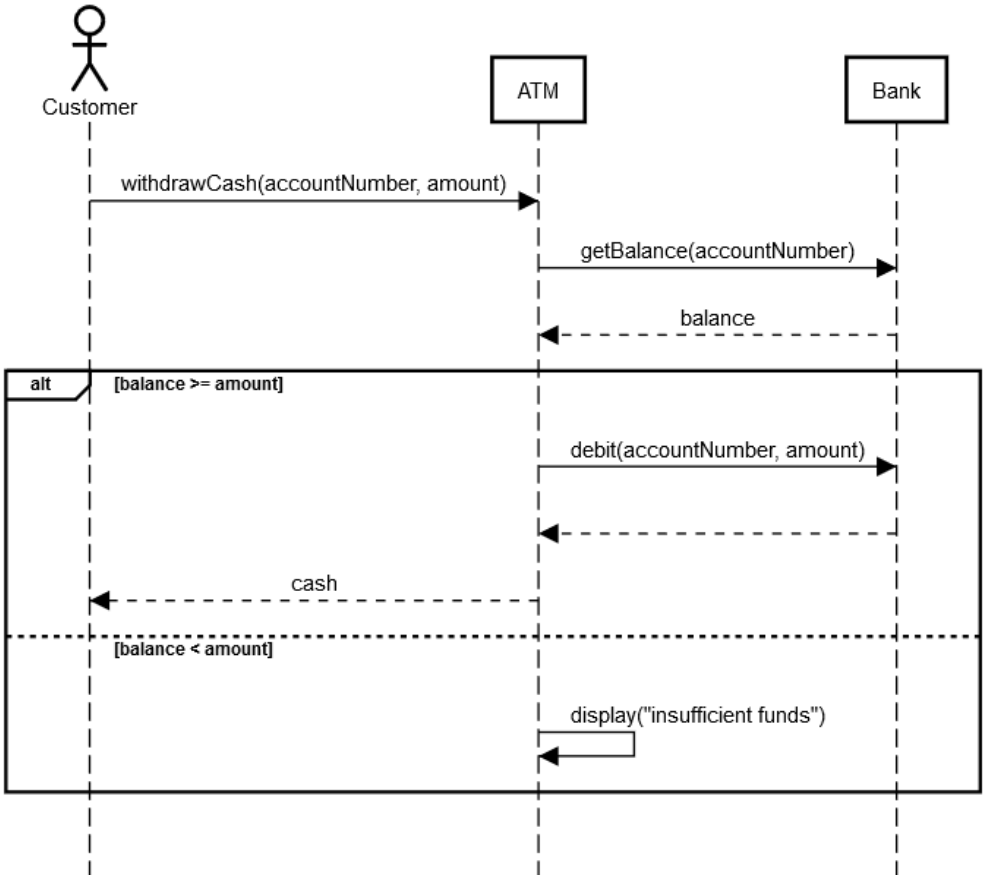


**Figure 1**

(a) Figure 1 is a type of *interaction diagram*.

(b) The vertical dashed lines are called *timelines*.

(c) Drawing the return arrows is optional.

(d) The `display` method may be either `private` or `public`.

(e) The `getBalance` method may be either `private` or `public`.

1.8 Given the following class, which of the statements concerning it are true?

(5 marks)

```
1  class A
2  {
3  public:
4      A(): a_{1}, b_{2}, c_{3} {}
5      A(int aa, int bb, int cc = -1): a_{aa}, b_{bb}, c_{cc} {}
6      A(int aa) { a_ = aa; b_ = 0; c_ = 0; }
7
8  private:
9      int a_ = 2;
10     int b_ = 2;
11     int c_ = 2;
12 };
```

(a) The constructor on line 4 is the class's default constructor.

(b) The constructor on line 4 makes use of in-class member initialisation.

(c) The constructor on line 6 first default-initialises the variables `a_`,`b_` and `c_` and then assigns them values.

(d) The following line of code

```
auto a3 = A{3,4};
```

results in `a_= 3`, `b_ = 4`, and `c_ = -1`.

(e) All of the constructors can be called without specifying a value for `c`.

[Total Marks Part A: 40]

# Part B

## Question 2

```
1  auto numbers = vector<int>{1,2,3,4,5,6,7,8};
2
3  for (auto z = begin(numbers); z != end(numbers); ++z)
4  {
5     // ?
6  }
7  cout << endl;
8
9  for (auto z = 0u; z != numbers.size(); ++z)
10 {
11    // ?
12 }
13 cout << endl;
14
15 for (auto z : numbers)
16 {
17    // ?
18 }
19 cout << endl;
```

**Listing 1:** Vector traversal

Answer the following questions with respect to Listing 1.

a) Give the *type* of the variable `z` that is deduced in each of the `for` loops. (3 marks)

b) Provide the missing code at lines 5, 11 and 17 so that each loop displays the contents of the vector. (3 marks)

c) Which of the above loops for displaying every element of a vector is to be preferred, and why? (2 marks)

[Total Marks 8]

## Question 3

a) Using raw (ordinary) pointers has a number of serious drawbacks. Name three issues which can occur when using raw pointers and for each issue provide a short code sample illustrating the problem. (9 marks)

b) Examine Listing 2 in the appendix and answer the following questions:

i) This program compiles and runs but has undefined behaviour in certain circumstances. Determine what these circumstances are and explain why the behaviour is undefined. (5 marks)

ii) Identify the particular line numbers in the `main` function at which the behaviour may be undefined. (2 marks)

iii) Rewrite the `LuckyPacket` class using a `unique_ptr` to `Toy` and provide additional functions which will ensure correct and consistent behaviour when it is used in `main`. (8 marks)

[Total Marks 24]

## Question 4

Tic-tac-toe (also known as noughts and crosses) is a game for two players, X and O, who take turns marking the spaces in a 3×3 grid. The player who succeeds in placing three of their marks in a horizontal, vertical, or diagonal row wins the game. The game ends in a draw if neither player is able to achieve a winning row after all spaces in the grid have been filled.

The following example game, as illustrated in Wikipedia, is won by the first player, X:

**Figure 2:** Tic-tac-toe

a) Provide the complete source code for a Tic-tac-toe game which allows two players to play against each other.

The game must be played in the *console*. You need to allow each player to take their turn by having them input the position at which they will make their mark, and you must display the updated game grid after each turn. Output a message when the game is over indicating who won, or if the game was a draw.

Your solution must be object-oriented and you should separate the presentation layer from the game logic.

(20 marks)

b) Provide a set of unit tests which verify that your solution works correctly. (18 marks)

[Total Marks 38]

[Total Marks Part B: 70]

(Exam Total: Four Questions – 110 marks : Full Marks – 100 marks)

**Please fill in the question numbers on the front page of your script.**

## Appendix: Source Code

```
class Toy { /*.....*/ };

class LuckyPacket
{
public:
   LuckyPacket(): toy_ptr_(nullptr) {
      if (rand() % 2) toy_ptr_ = new Toy{};
   }

   void open() {
      if (toy_ptr_ != nullptr) {
         cout << "Congrats, you won a toy!" << endl;
         // other functions which make use of the toy
         // ...
      }
      else cout << "Sorry, this lucky packet is empty." << endl;
   }

   ~LuckyPacket() {
      delete toy_ptr_;
      toy_ptr_ = nullptr;
   }

private:
   Toy* toy_ptr_;
};

int main()
{
   // seed the random number generator
   srand(static_cast<unsigned int>(time(0)));

   auto packet_1 = new LuckyPacket{};
   auto packet_2 = new LuckyPacket{*packet_1};
   auto packet_3 = LuckyPacket{};
   packet_3 = *packet_2;

   // use packet_1
   packet_1->open();
   delete packet_1;

   // use packet_2
   packet_2->open();
   delete packet_2;

   // use packet_3
   packet_3.open();

   return 0;
}
```

**Listing 2:** Lucky packet program