**Please fill in your Student Number and Name.**

**Student Number    : _____**

# University of Cape Town ~ Department of Computer Science

## Computer Science 1016S / 1011H ~ 2009

# November Exam

| Question | Max | Internal | External | Question | Max | Internal | External |
|----------|-----|----------|----------|----------|-----|----------|----------|
| 1 | 4 |  |  | 7 | 20 |  |  |
| 2 | 16 |  |  | 8 | 20 |  |  |
| 3 | 8 |  |  |  |  |  |  |
| 4 | 12 |  |  |  |  |  |  |
| 5 | 10 |  |  |  |  |  |  |
| 6 | 10 |  |  |  |  |  |  |
|  |  |  |  | **TOTAL** | **100** |  |  |

**Marks        : 100**

**Time          : 180 minutes**

**Instructions:**

   a) Answer all questions.

   b) Write your answers in pen in the spaces provided.

   c) Show all calculations where applicable.

## Question 1 [4]

Write the recursive function **calculate** that returns the $n^{th}$ number in the following sequence:

2  5  7  12  19  31  50 …

The first two numbers of this sequence – $F_0=2$ and $F_1=5$ - are given.

In general,

$F_{i+2}=F_i + F_{i+1}$

(This is very similar to the Fibonacci sequence.)                                    [4]

Note: No recursion means no marks.

```
public int calculate(int n)
{
  if (n==0)     [1]
    return 2;
  if (n==1)     [1]
    return 5;
  return (calculate(n-1)+calculate(n-2)); [2]
}
```

## Question 2 [16]

Examine the following Java application. It uses a Binary search algorithm to find the position of a key in an array.

```java
public class Exam1Demo {

    public static void main ( String [] args ) {
        int [] a ={2, 14, 21, 25, 33, 46, 52, 67, 71, 80};
        int result, no=52;
        System.out.println ("First Last Middle");
        result = search (a, 0, 9, no);
        if (result >= 0)
            System.out.println (no+" in element "+result);
        else
            System.out.println (no+" not found");
    }

    public static int search( int[]a, int first,
     int last, int item)
    {
        int result = 0;
        if (first > last)
            result = -1;
        else
        {
            int centre = (first + last)/2;
            System.out.println (first+" "+last +" "+ centre);
            if (item == a[centre])
                result = centre;
            else if (item < a[centre])
                result=search(a, first, centre-1, item);
            else if (item > a[centre])
                result=search(a, centre+1, last, item);
        }
        return result;
    }
}
```

a) What is the output from this program?                                          [4]

*First Last Middle*

*0 9 4*

*5 9 7*

*5 6 5*

*6 6 6*

*52 in element 6*

b) Rewrite the main method in the program above so that the integers for array **a[]** are read from a file named "exam.dat".                                          [8]

You may assume that "exam.dat" contains at most 50 integers; these integers are in strictly increasing numerical order; the file is terminated by the integer -1; and that all necessary libraries have been imported. The FileNotFoundException must be handled.

```
        public static void main ( String[] args )  {
```

*int[] a = new int[50];  // [1]*

*Scanner fileIn = null;*

*try {  // [1]*

  *fileIn = new Scanner (new FileInputStream("exam.dat"));  // [2]*

  *int i = 0;*

  *a[0] = fileIn.nextInt();*

  *while(a[i] != -1)*

  *{*

    *i++;*

    *a[i] = fileIn.nextInt();*

  *} // [2]*

```
            int result, no=52;
            result = Exam1Demo.search(a, 0, i-1, no);
            System.out.println("First Last Middle");
            if (result >= 0)
               System.out.println(no+" in element "+result);
            else
               System.out.println(no+" not found");
```

*}*

*catch (FileNotFoundException e)  // [1]*

*{*

  *System.out.println ("File exam.dat does not exist!");*

*}*

*fileIn.close( );  //[1]*

```
        }
```

c) Consider the situation where the file "exam.dat" did not have a terminator value of -1.  Now explain 2 different ways in which the end-of-file could be detected.  Discuss the advantages and disadvantages of each method.                                                [4]
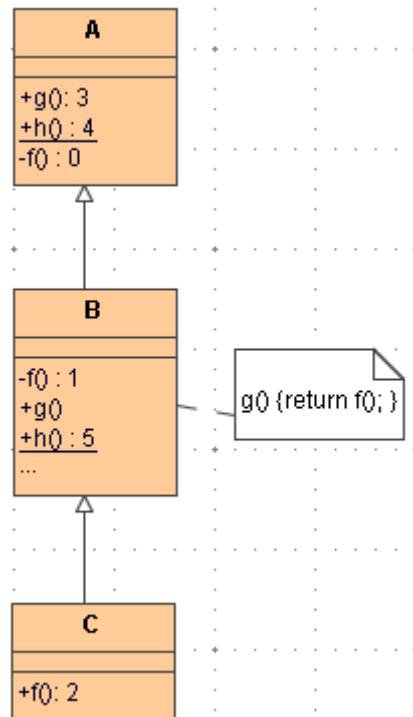
*1) Read past eof. End of file exception will be thrown & this can be caught.*

*2) Test inputStream for "hasNextLine( ) " and terminate loop when end detected*

*This is considered better as you have control.*

## Question 3 [8]

Use the following UML diagram to answer the questions that follow.  In the diagram, **g():3** implies that the member method named **g** returns a constant value of 3.



a) What principle of OOP design is illustrated in the inter-class relationships?  What are the exact relationships among each set of related classes?                                            [2]

*It is inheritance/generalization [1]*

*A is a super class and B is a derived class from A, C is a derived class from  B [1]*

b) What do each of the following accessibility modifiers in a UML class diagram indicate?     [2]

*+ : public*

*- : private*

*Underlined: static*

*Italics: abstract*

c) In the following sequence of statements that appears in a method in an unrelated class, indicate which statements will result in a compile-time error.  Then, assuming those statements are removed, indicate the value returned by each statement that produces a return value.       [4]

    i.   C ref1 = new A();

    ii.  A ref1 = new C();

    iii. ref1.f();

    iv. ref1.g();

    v.  ref1.h();

    vi. A ref2 = (A) ref1;

vii. ref2.g();

viii.ref2.h();

*ix.  C ref1 = new A(); False*

*x.   A ref1 = new C(); True*

*xi.  ref1.f(); False*

*xii. ref1.g(); 1*

*xiii.ref1.h(); 4*

*xiv.A ref2 = (A) ref1; True*

*xv.  ref2.g(); 1*

*xvi.ref2.h(); 4*

*[½ mark per question]*

## Question 4 [12]

Use the following program to answer the questions that follow.

```java
public class ASorter
{
    public static void sort(Comparable[] a, int numberUsed)
    {
        int index, indexOfNextSmallest;
        for (index = 0; index < numberUsed - 1; index++)
        {
            indexOfNextSmallest = indexOfSmallest(index, a, numberUsed);
            interchange(index,indexOfNextSmallest, a);
        }
    }

    private static int indexOfSmallest(int startIndex,
                                Comparable[] a, int numberUsed)
    {
        Comparable min = a[startIndex];
        int indexOfMin = startIndex;
        int index;
        for (index = startIndex + 1; index < numberUsed; index++)
            if (a[index].compareTo(min) < 0)//if a[index] is less than min
            {
                min = a[index];
                indexOfMin = index;
                //min is smallest of a[startIndex] through a[index]
            }
        return indexOfMin;
    }

    private static void interchange(int i, int j, Comparable[] a)
    {
        Comparable temp;
        temp = a[i];
        a[i] = a[j];
        a[j] = temp; //original value of a[i]
    }

}
```

a) What is the name of this sorting algorithm?                                      [2]

*Selection sort*

b) What is the complexity of the sort ?                                             [2]

*O(N^2)*

c) Explain how one would use this **sort** method with an array of Comparable type. [2]

*This is polymorphism. [up to 2, depending on explanation]*

*Objects of any class that implements Comparable interface can be plugged in for this sort. [2]*

d) List and explain the main differences between interfaces and abstract classes.  [2]

*Abstract Classes vs. Interfaces*

e) Why should one define an abstract method in an abstract class or an interface when its concrete subclasses implement the method anyway? Give a code example to illustrate your explanation.

[4]

Note: You may use the *withdrawal* method in the UML diagram below to answer the question.
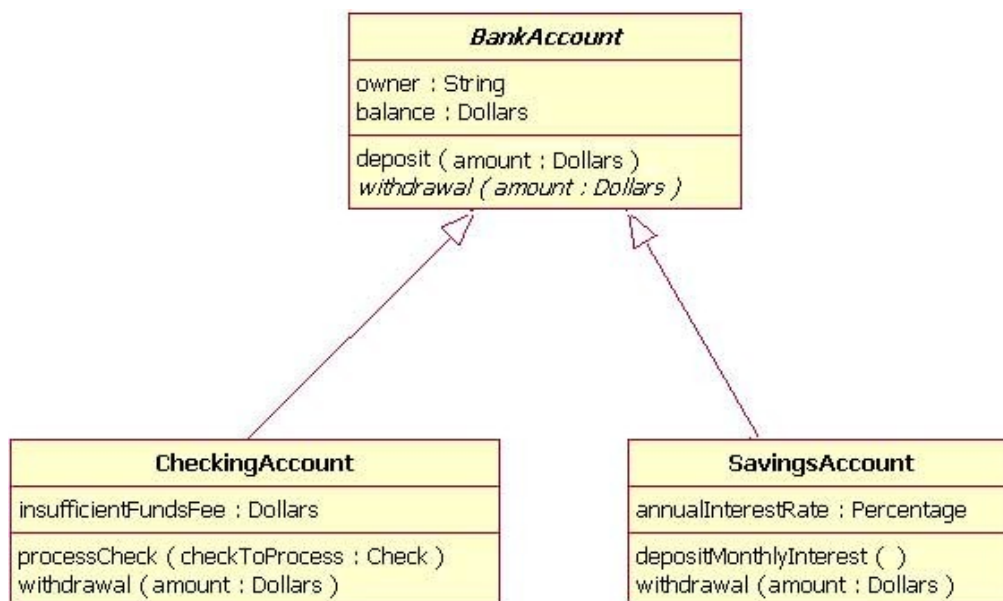
## Question 5 [10]

Use the following illustration of a doubly linked list and the code that follows to answer this question.
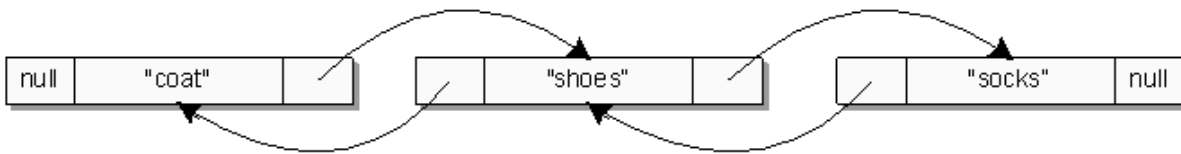


```
   public void delete()
   {
(i)   if (position == null)
         throw new IllegalStateException();            [2 marks]
      else if ((position.previous == null)
         && (position.next != null))
      {
(ii)                                                    [4 marks]
         head = head.next;
         position = head;
         head.previous = null;
      }
      else if (position.next == null)
      {
(iii)    position.previous.next = null;                 [3 marks]
         position = position.previous;
      }
      else
      {
         position.previous.next = position.next;
(iv)     position.next.previous = position.previous;    [1 mark]
         position = position.next;
      }
   }
```

a) Explain what the conditions are in each of the four cases labeled (i) to (iv). That is, state where we are on the list and what the special circumstances are at that point that must be considered when deleting.

*(i) The list is empty*

*(ii) We are at the first node and there are other nodes on the list.*

*(iii) We are at the last node and there are other nodes on the list.*

*(iv) We are at an internal node, i.e. there is at least one node before it.*

b) Explain what the code achieves in cases (i) to (iii) whenever the associated condition is true. Your explanation should answer the following questions: Which is the current node after the method executes? Which nodes are the predecessor and successor of this current node? What changes, if any, occurred in other nodes?

*(i) We throw an exception because we shouldn't be allowed to delete from an empty list.*

*(ii) We have the second node point to null as its predecessor. The former second node becomes the current node as well as the head. The former head node is lost as nothing points to it any longer.*

*(iii) We have the node before the last changed. Its successor becomes null.  It becomes the current node. The former last node is lost as nothing points to it any longer.*

## Question 6 [10]

Use the code below to answer this question.

```java
import javax.swing.*;
import java.awt.*;
import java.awt.event.*;

public class Events extends JFrame implements ActionListener,
WindowListener
{
    public static void main(String[] args)
    {
        Events gui = new Events();
        gui.setVisible(true);
    }

    public Events()
    {
        setTitle("Events Demo");
        setDefaultCloseOperation(JFrame.DO_NOTHING_ON_CLOSE);
        setSize(300, 200);
        setLayout(new FlowLayout());
        addWindowListener(this);

        JButton b = new JButton("Click");
        b.addActionListener(this);
        add(b);
    }

    public void actionPerformed(ActionEvent e)
    {
        System.exit(0);
    }

    public void windowOpened(WindowEvent e)
    {}

    public void windowClosed(WindowEvent e)
    {}

    public void windowClosing(WindowEvent e)
    {
        System.out.println("Ouch! Click somewhere else");
    }

    public void windowIconified(WindowEvent e)
    {
        System.out.println("I feel very small.");
    }

    public void windowDeiconified(WindowEvent e)
    {}

    public void windowActivated(WindowEvent e)
    {}

    public void windowDeactivated(WindowEvent e)
    {}
}
```
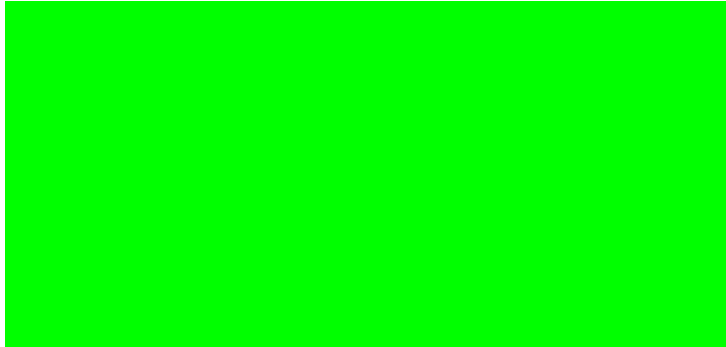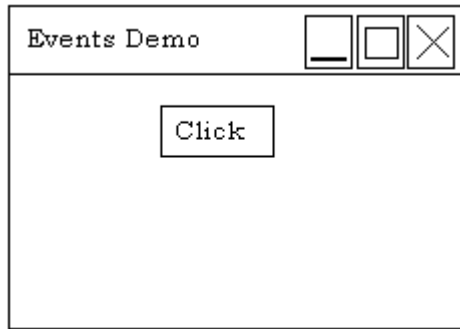
a) Draw the GUI that results from running the program. [3]



b) What happens when you click on the X button at the top right hand corner? [1]

*"Ouch! Click somewhere else" is printed out on the console and the window does not close.*

c) What happens when you click on the minimise button at the top right hand corner? [1]

*The words "I feel very small" appear on the console and the window is minimized.*

d) What happens when you click on the button labelled "Click". [1]

*The program ends.*

e) List all the changes that you would make to the program in order to use the **WindowAdapter** class rather than the **WindowListener** interface. [4]

*Do not implement the WindowListener interface but instead create an inner class which extends the WindowAdapter class. Have this inner class override the windowClosing method and the windowIconified method. Leave out the remaining 5 methods. Use an object of this inner class as the argument to the addWindowListener call.*

**Question 7 [20]**

a) Action based ethical theories can either be based on how well they conform to rules and norms (**deontological** ethical theory) or on what the outcomes are (**teleological** ethical theory). Please explain the main differences between these two different approaches. Can they ever result in different ethical decisions? [6]

*deontology: actions are essentially right or wrong, without regard to their consequences, Some actions are never justified no matter how good the outcome — ends cannot justify means. Can also give examples of duties (keep promises, right wrongs, …) or rights (privacy, property …)[3]*

*teleology: When actions are judged morally right based upon their consequences, sometimes means will justify by the ends, Correct actions are those that produce the most good or optimise the consequences of choices, [3]*

*So yes they can result in different decisions and this can revolve around the issue of the means used to achieve the end [1] (obviously only 6 marks max)*

b) Should network managers be licensed? Discuss the advantages and drawbacks? [4]

*Advantages: (max [2])*

*a) Qualified people more likely to do a good job. [1]*

*b) Assured that an appropriate methodology is used. [1]*

*c) Assured that licensee has appropriate experience (you'll have to be apprenticed before getting license). [1]*

*Drawbacks: (max [2])*

*a) Competent people, without degrees or accreditation can be excluded from practicing. [1]*

*b) Small projects - less need for professionals, licensing would be overkill. [1]*

*Anything else that is relevant.. [1]*

c) How is Richard Stallman's idea of "copyleft" related to copyright? What does copyleft set out to achieve?. [4]

*Uses copyright law, but flips it over to serve the opposite of its usual purpose: instead of privatizing software, it becomes a means of ensuring it is kept free and that such freedom is inalienable. [2] It gives everyone the right to use the programs and modify it but not the permission to put restrictions on it. [2].*

d) What sorts of Information and Communications Technologies are appropriate in the developing world? Give two (2) examples and justify each of them by saying why you think it is the right technology. [6]

*Number of choices, [1] for the technology and [2] for the justification. Examples (may have others):*

*Wireless networks — no require physical landlines, no expensive last-mile infrastructure*

*Mobile devices (cellphones) — widespread, less expensive and easier to use than PCs, batteries*

*Voice over IP (VoIP) — doesn't require sophisticated telecommunications infrastructure*

*Open Source Software — Cost-effective and can be customised to local needs.*

## Question 8 [20]

**Matric Results Kiosks**

Information kiosks are desktop computers placed in secure locations for members of the public to access various forms of information.

The government has decided to use existing kiosks in rural areas to distribute matric examination results. You have been hired as a consultant to develop additional software for these kiosks to enable this new function.

The results are stored on disk in a file named "results.txt", sorted on the subject name, with the following format:

```
StudentIDX SubjectA 56
StudentIDY SubjectA 69
StudentIDZ SubjectA 80
StudentIDX SubjectB 75
StudentIDZ SubjectB 72
```

a) Write a Java method to read the contents of this file into an array of objects of type Entry. Assume there are at most 5000 entries in the file and the student IDs and subjects do not contain spaces. [5]

```java
class Entry
{
   public String studentID;
   public String subject;
   public int mark;
}
```

*Entry [] a = new Entry[5000];*

*try {*

  *fileIn = new Scanner (new FileInputStream("results.txt"));  // [2]*

  *int i = 0;*

  *while (fileIn.hasNext ())*

  *{*

    *a[i] = new Entry ();*

    *a[i].studentID = fileIn.next();*

    *a[i].subject = fileIn.next();*

    *a[i].mark = fileIn.nextInt();*

  *}*

*fileIn.close ();*

b) After the file has been loaded into memory, multiple students can look up their results by entering their studentIDs. Why is this process slow? What algorithms would you implement in your code to make looking up of results more efficient? [5]

Note: Do NOT write code – simply describe the algorithms you would use.

*slow because the entire array has to be searched each time O(n).*

*to make it faster, presort the array on studentID – O(n Log n), then use a binary search – O (Log n) of the sorted array for every query and scan linearly to get all subject entries.*

c) Sketch a user interface for your application and explain how a user will interact with this application. [5]

*any reasonable interface with student id entry and space to display results*

d) Discuss 2 major professional issues that arise in designing this application. [5]

*Privacy: students should not be able to view the results of other students in detail*

*Security: the data should be safe from tampering*

*Reliability: users of the data should have some assurance of accuracy*

*etc.*