Name:

____

____

Student Number:

____

Please fill in your Student Number and Name.

Student Number : ______________________

University of Cape Town ~ Department of Computer Science

Computer Science 1015F ~ 2008

January Exam

Question	Max	Internal	External	Question	Max	Internal	External
1	9			7	3		
2	8			8	8		
3	8			9	15		
4	15			10	10		
5	10						
6	14						
				TOTAL	**100**		

Marks : 100

Time : 180 minutes

Instructions:

a) Answer all questions.

b) Write your answers in the space provided.

c) Show all calculations where applicable.

Question 1 [9]

a) What is the purpose of an operating system? [1]

manage the resources of a computer

b) Give 2 examples of operating systems (from different companies or communities). [2]

Windows XP [1], Linux [1], MacOS [1], ...

c) What is the purpose of random-access memory? [1]

store data temporarily while the computer is powered on

d) What is an algorithm? [1]

set of instructions to perform a task

e) List 2 reasons why algorithms must be precise. [2]

repeatable [1] can be executed by different people [1] predictable [1]

f) Briefly describe 1 advantage and 1 disadvantage of using Java bytecode, as opposed to machine code. [2]

ad: portability [1] disad: slower[1] more work [1]

Question 2 [8]

Consider the following program and answer the questions that follow.

```
import java.util.Scanner;

class Test
{
   public static void main ( String [] args )
   {
      Scanner input = new Scanner (System.in);

      int a = input.nextInt();
      int b = input.nextInt();
      int c = input.nextInt();

      int x = Math.min (Math.min (a, b), c);
      int y = Math.max (Math.max (a, b), c);

      System.out.println ((a+b)/6.0f+(b+c)/6.0f+(a+c)/6.0f);
   }
}
```

a) What does this program do? [2]

calculates the average of 3 integers

b) What is the output if the input is the numbers 5, 3 and 7? [1]

5

c) List all methods (other than main) from the program. [1]

println,min,max,nextInt

d) Why must the main method be static? [1]

it is invoked by the JVM without an instance being created

e) Rewrite the first statement of the main method in such a way that there is a syntax error. [1]

Anything syntactically invalid: e.g., Scanner input new Scanner (System.in);

f) In the context of a debugger, what is a breakpoint? [2]

a specified line of code at which the debugger will stop when executing the program in debugging mode

Question 3 [8]

a) Briefly describe an algorithm to take a taxi or bus to town. Assume you are at the bus/taxi stop. There should be at most 6 steps. [3]

wait for correct bus, get onto bus, pay driver, sit down, wait for stop, get off bus

b) Write the Java statement to input the number of taxis that pass you into the variable **N**. You may assume **N** is already declared as an int and there is already a Scanner object named **input**. [2]

N = input.nextInt();

c) Write the Java statement to calculate the variable **totalTime** (in seconds) as the time you wait for **N** taxis to pass if one taxi passes every **timePerTaxi** seconds. You may assume **totalTime** and **timePerTaxi** are already declared as float variables and **N** is declared as an int. [1]

*totalTime = N * timePerTaxi;*

d) Write the Java statement to output "Time wasted at bus stop: ", immediately followed by the value of the variable **totalTime**. [2]

System.out.println ("Time wasted at bus stop: " + totalTime);

Question 4 [15]

Examine the following code and answer the questions that follow.

```
public class Driver {
   public static void main (String[] args) {
      for ( int date=0; date<10; date++ ) {
         if (date%2 == 0) {
            System.out.println(convertDate(date));
         }
      }
   }
   // convert a number to a day of the week
   public static String convertDate(int number) {
      String date = "";
      if (number == 2)
         date = "Monday";
      if (number == 3)
         date = "Tuesday";
      if (number == 4)
         date = "Wednesday";
      if (number == 5)
         date = "Thursday";
      if (number == 6)
         date = "Friday";
      if (number == 7)
         date = "Saturday";
      if (number == 8)
         date = "Sunday";
      return date;
   }
}
```

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

Monday

Wednesday

Friday

Sunday

b) Rewrite the convertDate method, converting the *if-else* statement to an equivalent *switch* statement. [4]

```
public static String convertDate(int number) {
  String date = "";
  switch (number) {
  case 2:
    date = "Monday";
    break;
  case 3:
    date = "Tuesday";
    break;
  case 4:
    date = "Wednesday";
    break;
  case 5:
    date = "Thursday";
    break;
  case 6:
    date = "Friday";
    break;
  case 7:
    date = "Saturday";
    break;
  case 8:
    date = "Sunday";
    break;
  }
  return date;
}
```

c) What is the result of each of the following expressions? The result will be either true or false. Assume variable **`week = 7`**. [2]

i. **`week > 6 || week == 5`**

True [1/2]

ii. **`week < 3 || week > 4`**

True [1/2]

iii. **`week > 3 && week < 0`**

False [1/2]

iv. `week >= 5 || week < 0`

True [1/2]

d) Write a Java method to print out a blank grid for a working week calendar, such as:

```
+ - + - + - + - +
|   |   |   |   |
+ - + - + - + - +
|   |   |   |   |
+ - + - + - + - +
```

Your method should have the header:

```
void printGrid ( int numberOfWeeks, int numberOfWorkingDays )
```

The above pattern is produced when invoking `printGrid(2,4).` [5]

```
public void printGrid ( int numberOfWeeks, int numberOfWorkingDays) {
  String out = "+";
  for (int j=0; j<numberOfWorkingDays; j++) {
    out += "-+";
  }
  String out_mid = "|";
  for (int j=0; j<numberOfWorkingDays; j++) {
    out_mid += " |";
  }
  for (int i=0; i<numberOfWeeks; i++) {
    System.out.println(out);
    System.out.println(out_mid);
  }
  System.out.println(out);
}
```

Question 5 [10]

a) If a class called **`Student`** is created and one of the data fields (or instance variables) for the **`Student`** class is a reference to a **`Date`** object, what is the relationship between the `Student` object and the `Date` object? [1]

i. The **`Date`** object is a **`Student`** object.

ii. The **`Date`** object has a **`Student`** object.

iii. The **`Student`** object is a **`Date`** object.

iv. The **`Student`** object has a **`Date`** object.

The Student object has a Date object.

b) Write skeleton Java code to illustrate the above description (i.e., create the Student class and a public Date object such as enrolmentDate). [2]

```
public class Student {
  public Date enrollmentDate = new Date();
  /* Other code here*/
}
```

c) Write the body of the main method of a driver program to create an object of the **`Student`** class, called **`aStudent`**. [1]

Student aStudent = new Student();

d) Can **`enrolmentDate`** be accessed by calling **`aStudent.enrolmentDate`** from the driver program? Why or why not? [1]

No. Because enrolmentDate was declared private

e) What do you need to change in the Student class so that the value of **`enrolmentDate`** can be accessed from the driver class? List two ways and write code to illustrate your methods. Explain which one is the better way. [5]

#1: enrolmentDate can be declared public in the Student class

public Date enrolmentDate = new Date();

#2: in the Student class, we can declare mutator and accessor methods to set and access value of enrolmentDate

```
private Date enrolmentDate = new Date();
public Date getEnrolmentDate() {
  return enrolmentDate;
}
```

#2 is a better way in terms of information hiding and data encapsulation

Question 6 [14]

a) Consider the Person class below, and add constructors, accessor and mutator methods as described by the comments below:

```
public class Person
{
    private String name;
    private int age;
    private double average_weight_gain;
```

// i) Constructor with three arguments that are used to initialize the instance variables [2]

```
public Person(String n, int a, double g)
{
  name = n;
  age = i;
  average_weight_gain = g;
}
```

// ii) Accessor methods for all the instance variables [3]

```
public String getName()
{
  return name;
}
public int getAge()
{
  return age;
}
public double getGain()
{
  return average_weight_gain;
}
```

// iii) Two mutator methods with one argument each to set the *name* and *age* instance variables [2]

```
public void setName(String n)
{
 name = n;
}
public void setAge(int a)
{
 age = a;
}
```

// iv) One mutator method with an array argument that is used to calculate the
// *average_weight_gain* instance variable, returning no value [3]

```
public void calculateAverageGain (double [] g)
{
 double sum = 0.0;
 int i;

 for (i=0; i < g.length; g++)
   sum = sum + g[i];
 average_weight_gain = sum / g.length;
}
```

```
    }
```

a) Consider the driver class below, and add a statements to create an object called **aPerson** of type **Person** for "Peter", age 21, with an average monthly weight_gain of 0.0kg, using your constructor above. Use your mutator method above to calculate the average weight gain of the person over the festive period from November to February - his weight gains were 0.1 kg, 1.2 kg, 0.5 kg and -1.2 kg. Display the name, age and average weight gain for the person using your accessor methods above. [4]

```
public class ConstructorsDemo
{
    public static void main (String args[])
    {
Person aPerson = new Person("Peter",21,0.0);
double [] gain ={0.1, 1.2, 0.5, -1.2};
aPerson.calculateAverageGain(gain);
System.out.println ("Name = " + name + aPerson.getName()+ "Age = " +
aPerson.getAge() + "Average Weight Gain = " + aPerson.getGain();
        }
    }
```

Question 7 [3]

a) How can arrays be tested to check if their content is identical? [3]

The best way to test two arrays to see if the contents are the same is to write a method that accomplishes the task. This method would iterate through each array comparing indexed variables. If arrays are tested for equality using the == operator this only checks if it is the same array.

Question 8 [8]

The following set of methods sorts an array of numbers in descending order. Complete the missing methods as indicated by the comments.

```
/**
Precondition:  The array has values.
Action:Sorts a so that a[0] >= a[1] >= ... >= a[a.length-1]
                                                              [4]
*/
public void selectionSort ( int[] a )
{
int indexOfLargest;
for ( int i=0; i<a.length; ++i )
{
indexOfLargest = LargestIndex (i, a);
swap (i, indexOfLargest, a);
}
}

/**
Returns the index of the largest value among
a[start], a[start + 1], ... a[a.length-1]                     [4]
*/
private int LargestIndex ( int start, int[] a )
{
 int max = a[start];
 int indexOfMax = start;
 for(int i=start + 1; i < a.length; i++)
 {
  if(a[i] > max)
  {
   max = a[i];
   indexOfMax = i;
  }
 }
   return indexOfMax;
}
```

```
/**
Precondition:  i and j are legal indices for the array a.
Postcondition:  Values of a[i] and a[j] have been
interchanged
*/
private void swap ( int i, int j, int[] a )
{
   int temp;
   temp = a[i];
   a[i] = a[j];
   a[j] = temp;
}
```

Question 9 [15]

a) Match each of the following terms to the concepts below: [2]

overriding, overloading, inheritance, composition

inheritance = variables and methods automatically exist in the subclasses of a base class

overriding = methods in base class and subclass with same signature but different code

overloading = methods in base class and subclass with same name but different signature

composition = a class has an instance variable which is itself an Object [½ x 4]

b) Suppose that a 2-dimensional array `sales` has been declared as below.

```
int[ ][ ] sales ;
```

Give Java code to fill `sales` with 120 zeroes: there must be 100 rows each with 12 elements, and every element must be initialised to 0. [4]

```
  sales = new int[100][12];
  for (int j=0; j<sales.length; j++)
     for (int k=0; k<sales[j].length; k++)
        sales[j][k] = 0;
```

c) Suppose your program has changed the values in `sales` to contain monthly sales data from 100 shops. Give Java code to print monthly totals – i.e., print the sum of the values in each column of `sales`. The 12 values should appear one after the other on the same single line of output, separated by spaces. [4]

```
int sum;
  for (int j=0; j<12; j++)
     {   sum = 0;
        for (int k=0; k<sales.length; k++)
           if (j < sales[k].length)    //not expected to check this tho
              sum = sum + sales[k][j];
       System.out.print(sum + " ");
     }
  System.out.println();
```

d) Consider the classes below, then list the output of the `main` program. [5]

```
public class Person
{
   protected String name;
   protected String address;
   public Person(  )
   {   name = "unknown"; address = "N/A";
       System.out.println("Created unknown person");
   }
   public Person( String pname, String adr )
   {   name = pname; address = adr;
       System.out.println("Created person");
   }
}

public class Child extends Person
{
    int age;
    public Child  (String cname, String adr, int years )
    {   super ( cname, adr );
        age = years;
        System.out.println("Created child");
    }
    public Child (  )
    {   age = 0;
        System.out.println("Created unknown child");
    }
    public String toString( )
    {
        return (name + " ( " + address + ") aged " + age));
    }
}

public class Test
{
   public static void main ( String args[ ] )
   {
      Child ann = new Child( "ann", "UCT", 6);
      Child kiddie = new Child(  );
      System.out.println( kiddie );
   }
}
```

Output from running `Test` is:

Created person

Created child

Created unknown person

Created unknown child

unknown (N/A) aged 0

Question 10 [10]

a) Convert the binary number `101010`$_2$ to hexadecimal. [1]

2A

b) Convert the hexadecimal number `CAB`$_{16}$ to binary. [1]

110010101011

c) Convert 43_{10} to binary (i.e., convert the decimal number 43 to base 2). Show your working. [2]

101011

d) Convert 3.375_{10} to binary (i.e., convert the decimal number 3.375 to base 2). Show your working. [3]

11.011

e) What is the 8-bit *two's complement* binary representation of the decimal number -13_{10}? Show your working. [3]

11110011