CITS1001 exam 2017
Official cover page to be substituted here

June 15, 2017

WITH SAMPLE SOLUTIONS

This page has been left intentionally blank

Question 1 Class Definitions (10 marks)

1a) Write a Java class called `Property` to represent a property such as a house or apartment for a property rental business. Include brief comments in your class definition as necessary, but full Javadoc is *not* required. The class should have:

- four fields that capture the property's street address, its rental price per month, the number of bedrooms, and whether the property has off-road parking; and
- a constructor that initialises the fields;
- a mutator method for setting the number of bedrooms field; it should check for a reasonable input value;

1b) Write a Java class `Rentals` for managing a group of properties that are available for rental. This class should have:

- one field to store all the currently available properties;
- a constructor that initialises this field;
- a method that takes a property as argument and removes it from the list;
- a method that takes two properties as arguments and returns the property with the lowest monthly rent.

Use the following pages for your answer to this question

Marking guide: check for sensible choice of names and types eg double possible for monthlyRent. Allow for 0 bedrooms and maybe an upper bound too. Do not allow negative bed count.

Answer to Question 1a here

```
public class Property {
   String address;
   int monthlyRent;
   int numberOfBedrooms;
   boolean offRoadParking;

//no error checking here
public Property(String address, int monthlyRent,
    int numberOfBedrooms, boolean offRoadParking) {
    this.address = address;
    this.monthlyRent = monthlyRent;
    setNumberOfBedrooms(numberOfBedrooms);
    this.offRoadParking = offRoadParking;
}

public void setNumberOfBedrooms( int num ) {
    if ( numberOfBedrooms >= 0 ) {
      numberOfBedrooms = num;
   }
}
}
```

Answer to Question 1b here

```
public class Rentals {
  ArrayList<Properties>  currentRentals;

 public Rentals {
     currentRentals = new ArrayList<>();
 }


 public void removeProperty( Property p ) {
      currentRentals.remove(p);
 }

 public Property cheapest(Property p1, Property p2) {
     if (p1.getMonthlyRental() <= p2.getMonthlyRental()) {
        return(p1)
     } else {
         return(p2);
     }
 }

}
```

Question 2 Conditionals (10 marks)

Complete the three segments of code marked `//TODO` below. Your methods should throw suitable exceptions if their parameter values are unsuitable. Implement the methods `emptyMachine` and `calculateFare` according to their Javadoc specifications.

```
public class TicketMachine {

   private int price;    //The price of a ticket.
   private int balance;  //The amount of money entered so far.
   private int total;    //The total amount of money collected.

   public TicketMachine( int price )  {
       //TODO
    }

    /** Simulate emptying the machine of money by
    * resetting the total to 0.
    * @return int amount of money that was in the machine
    * before it was reset.
    */
    public int emptyMachine()  {
        //TODO
    }

    /** Calculate a fare based on the standard ticket price,
    *   the zone of travel and a discount rate for special kinds
    *   of travellers (eg senior, school fare etc).
    *   @param boolean innerZone; if false the fare is 50% higher
    *   @param int discount percentage to reduce the fare by
    *   @return int fare, the calculated discounted price
    *   rounded to the nearest integer value.
    */
    public int calculateFare( boolean innerZone, int discountPct ) {
         //TODO
     }
}
```

Answer question 2 here

```
    public TicketMachine( int price )  {
        if (cost <= 0) {
            throw new IllegalArgumentException(
                "Negative price not permitted");
        } //otherwise initialise
        this.price = price;
        balance = 0;
        total = 0;
    }

    public int emptyMachine()  {
        int  beforeEmpty;
        beforeEmpty = total;
        total = 0;
        return (beforeEmpty);
    }

    // ensure: there is a strategy to do the correct rounding,
    // not just int divide
    // ensure: the fare is returned, rather than the price changed
    public int calculateFare( boolean innerZone, int discountPct )
    {
        double fare;
        if (discountPct < 0 || discountPct > 100) {
            throw new IllegalArgumentException(
                "discountPct must be in the range 0 to 100");
        }
        if ( innerZone ) {
            fare = price - 1.0*price*discountPct/100;
        } else { //zone is outer
            fare = 1.5 * price;
            fare =  fare - fare*discountPct/100;
        }
        return Math.rount(fare);
    }
}
```

Answer question 2 here if needed

Question 3 List Collections (10 marks)

```
public void mystery3( ArrayList<Integer> list ) {
    for (int i = 0; i < list.size(); i++) {
        int element = list.get(i);
        list.remove(i);
        list.add(0, element + 1);
    }
}
```

3a) Fill in the table below to show the value of `list` after `mystery3` (above) is executed with a `list` parameter containing the given elements. Answer Question 3a in the table

`list` input	final contents of `list`
$10, 20, 30$	
$8, 2, 9, 7, 4$	
$-1, 3, 28, 17, 9, 33$	

$10, 20, 30$	31, 21, 11
$8, 2, 9, 7, 4$	5, 8, 10, 3, 9
$-1, 3, 28, 17, 9, 33$	34, 10, 18, 29, 4, 0

3b) Write a short description to summarise what this method does.

Answer Question 3b here

It adds 1 to each element and reverses the list.

Question 4 Map Collections (10 marks)

4a) Declare a field variable to represent a phone book that stores the names and phone numbers of a collection of people. (2 marks)

Answer Question 4a here

```
private Map<String, String> phonebook;  //or maybe
 private HashMap<String, String> phonebook;
```

4b) Write a method called `lookupNumber` that returns the phone number of a given person in the phone book. (3 marks)

Answer Question 4b here

```
public String lookupNumber ( String person ) {
    return (phonebook.get(person));
}
```

4c) Now consider a phone book *in reverse*. That is, given a phone number, finding the name of the person it belongs to. Write a method called `personWithNumber` that takes a phone number, and returns the name of a person with that number. If there are multiple people, then return the name of the first person you find; if no people have that number, just return `null`. (5 marks)

Answer Question 4c here

```
public String personWithNumber ( String number ) {
    Set<String> people = phonebook.keySet();
    for (String person : people) {
        if (phonebook.get(person).equals(number)) {
            return (person);
        }
    }
    return null; //if none found
}
```

Question 5 Testing (10 marks)

A method `uwaSemesters` is required that takes two parameters representing a day and a month and returns a String indicating that type of activity for that date. An activity is one of: `"Classes"` (during semester), `"StudyBreak", "Examinations"` or `"Vacation"` as determined by the calendar below. The method only has to handle first semester dates for this question. Assume that months are specified as an integer between 1 and 12 (1 for January, 2 for February, and so on) and that the day of the month is a number between 1 and 31. You are *not* required to check for invalid month, day combinations such as month=2 and day=30.

We will first write the test cases and then the Java code.

February 27	First Semester classes commence
April 17 - 21	Non-teaching study break
June 2	First Semester classes end
June 5 - 9	Pre-examination study break
June 10 - 24	First semester examinations
June 26 - July 30	Student Vacation

5a) Add four (4) new test cases to the table below for testing the correctness of an implementation of this method. The first line (in italics) is given as an example. Include a brief reason for each test case in the Rationale column. (4 marks)

Answer Question 5a here

day	month	Expected output	Rationale
1	*4*	*"Classes"*	*normal case*

5b) Write Java code to implement the `uwaSemesters` method. State any assumptions that you make. Marks will be awarded for concise coding. (6 marks)

Answer Question 5b here

```
        5a) 1 mark for each good test (of 4).  For example,
Look for good coverage of the test cases (given only 4 are offered)
                                  :
       e.g.  normal, boundary, exception cases and error case
Include error out of bound, but not illegal month, day combinations
                  season(27, 2) "Classes" boundary
                 season(9, 6) "StudyBreak" boundary
                   season(10, 5) "Classes" normal
                   season(1, 7) "Vacation" normal
                 season(13, 6) "Examinations" normal
                   season(1, 1) "Vacation" assumed
                season(-1,9) "Error" illegal arguments
  5b) 7 marks:  1 for correct header, 2 for correct boolean logic
  including illegal cases, 2 for error case and structure - think
         through the choices rather than listing all cases
Students may organise the cases in different order, eg "Classes" as
the fall through.  And either 7 or 12 is OK as upper bound for month
                             errors.
```

```
public String uwaSemesters(int day, int month)
{
    if ( month<1 || month >12 || day<1 || day>31)
        { throw new IllegalArgumentException (
                  "Month or day out of range"); }
    if ( (month==6 && day>=10 && day<=24)
       { return "Examination"; }
    if (month==4 && day>=17 && day<=21)
       { return "StudyBreak"; }
    if ((month==2 && day>=27) || (month>=3 && month<=5) ||
        (month==6 && day<=2))
       { return "Classes"; }
    else
      { return "Vacation"; }
}
```

Question 6 Debugging (10 marks)

The program below is intended to produce the output, `a is the smallest!`. But the program contains at least 7 errors. Mark the errors on the code below. Additionally, for each error you identify, suggest a way to correct it.

Answer Question 6 here and on the next page

```
public class Oops{

public static void main(String[]args){
        int a = 7,b = 42;
        minimum(a,b);
        if {smaller = a} {
           System.out.println("ais thesmallest!");
        }
     }

     public static void minimum(int a,int b){
        if (a < b){
            int smaller = a;
        } else (a => b) {
            int smaller = b;
        }
        return int smaller;
    }
}
```

Any of these:

line	error	fix
5	need a var to return to	`int smaller = minimum(a,b);`
6	use () not for expression	`if (smaller == a)\verb`
6	use == for comparison, not =	ditto
7	missing space	`System.out.println("ais the smallest!");`
11	return type should not be void	int in signature
13,15	only declare smaller once	int smaller; after line 12
14	no test for else	remove `(a=>b)`
14	wrong syntax for greater equal	`(a>=b)`
17	dont include type in return	return smaller;

Continue answering Question 6 here if needed

Question 7 Algorithms (10 marks)

Some cognitive psychologists believe that people recognize words based on their shape. For this reason most people can make sense of the following mis-spelled text.

> It de'osnt mttaer in waht oredr the ltteers in a wrod are, the olny iprmoetnt tihng is taht frist and lsat ltteer is at the rghit pclae. The rset can be a toatl mses and you can sitll raed it wouthit porbelm. Tihs is bcuseae we do not raed ervey lteter by itslef but the wrod as a wlohe.

Suppose you are asked to write a Java program to help test this hypothesis. Write a method `scramble` that takes a single word as input and returns a string with the internal letters in random order but with the first and last letter unchanged. Write helper methods as required and state any assumptions you make.

Answer Question 7 here or over the page

```
public class Scrambler
{
    // helper swaps array elements i and j
    public static void exchange(char[] a, int i, int j) {
        char temp = a[i];
        a[i] = a[j];
        a[j] = temp;
    }

    // return a random integer between 0 and n-1
    public static int uniform(int n) {
        return (int) (Math.random() * n);
    }

   // ... continued
```

Answer Question 7 here

```
    //strategy 1:  take an String of characters and
    //swap middle chars at random
    public static String scramble(String word) {
        // put all string characters into an array
        char[] newword = word.toCharArray();
        int n = newword.length;
        // do random swaps between any positions except the first and last
        for (int i = 1; i < (n-1); i++) {
            int r = i + uniform(n-1-i);   // between i and n-2
            exchange(newword, i, r);
        }
        String str = ""+newword[0];
        for (int i=1; i<n; i++) {
            str = str + newword[i];
        }
        return str;
    }

    // alternative strategy: save first, put middle chars into a bag,
    // get them out one at a time at random, finally append the last char
    public static String scramble2(String word) {
        int n = word.length(), randompos = 0;
        ArrayList<Character> middle = new ArrayList<>();
        for (int i=1; i<(n-1); i++) {
            middle.add((Character)word.charAt(i));
        }
        //now build a scrambled string
        StringBuilder newword = new StringBuilder();
        newword.append(word.charAt(0));
        while (middle.size() > 0) { //pull letters out one at a time
            randompos = (int) (Math.random() * middle.size());
            newword.append(middle.get(randompos));
            middle.remove(randompos);
        }
        newword.append(word.charAt(n-1));  //put the final letter in place
        return (newword.toString());
    }
}
```

Question 8 Code Quality (10 marks)

The methods `findMostSocialAAA` and `findMostSocialBBB` are intended to search an `ArrayList<Person>` object called `contacts` for the `Person` object with the highest activity level. Assume the `Person` class has a `getTotalActivity` method.

Compare and contrast these two methods. Comment on the strengths and weakness of each implementation, using the three criteria of: *correctness, design*, and *readability*. Mark up the code with your observations on its quality and then summarise your assessment against the given criteria.

```
public Person findMostSocialAAA() {
     int maxActivity = 0;
     int pActivity;
     Person maxPerson = null;
     for (Person p : contacts) {
         pActivity = p.getTotalActivity();
         if (pActivity <= maxActivity) {
             maxActivity = pActivity;
             maxPerson = p;
         }
     }
     return maxPerson;
}

public Person findMostSocialBBB() {
      Person mp;
      int ma = 0;
      Iterator<Person> it = contacts.iterator();
      while (it.hasNext()) {
        Person p = it.next();
        int a = p.getTotalActivity();
        if (a > ma) {
            ma = a;
            mp = p;
         }
     }
    return mp;
}
```

Continue answering Question 8 here

Sample answer:
1 mark per reasonable point,
organisation should address the four criteria,
mark up the code and reference it in the answer
mark 8+ excellent, 5+ good, 3+ poor 0+ very poor

Spare page for working