Your name: ________________________________

Data Structures and Algorithms for Language Processing

Midterm Exam WS 2011/2012
Lecturer: Verena Henrich

December 19, 2011

INSTRUCTIONS

This exam contains 13 questions worth a total of 40 points.

You can earn up to 1 point by answering the bonus question correctly, but you cannot earn more than 40 points on this exam.

For all program fragments, assume any necessary "imports" are present.

Write your name at the top of each page and on any extra sheets of paper that you use.

Turn in all pages of the exam and any extra sheets of paper that you use.

Good Luck!

Score: _______ / 40

Your name: ________________________________

Part I – Short Answer

Question 1 (3 points)

For each of the javadoc comments below, write the **method headings only**. Choose meaningful names for the methods. Use doubles for numerical values. Do not implement the methods.

```
/**
 * Transfer amount from this account to otherAccount.
 * @param otherAccount the account to tranfer money to
 * @param amount the amount to transfer
 */
```

```
/**
 * Get an array containing the synonyms of this word.
 * @return an array containing synonyms
 */
```

```
/**
 * Determine if name is contained in list.
 * @param name the name to search for
 * @param list an array of names
 * @return true if name is contained in list, false otherwise
 */
```

Your name: ________________________________

Question 2 (2 point)

`MyClass` doesn't compile – explain why.

```
public class MyClass
{
    private String name;
    private String email;

    public MyClass(String aName)
    {
        name = aName;
    }

    public MyClass(String aName, String anEmail)
    {
        name = aName;
        email = anEmail;
    }

    public MyClass(String anEmail)
    {
        email = anEmail;
    }
}
```

Your name: ________________________________

Question 3 (4 points)

Fill in the blanks:

The qualifier `private` should always be used when declaring _______________________.

A(n) _______________________ method can NOT access any instance variables.

When two or more methods within a class definition have the same name, they are said to be

________________________.

_______________________ errors are not caught by the compiler and do not cause a program

to exit, but cause a program to give incorrect results.

A(n) ________________________ is a special kind of method used to initialize instance

variables.

An integer that indicates a position in an array is called a(n) _________________.

A mixture of spoken language and computer language used to express an algorithm is called

_________________________.

_______________________ methods within a class definition can only be called by other

methods within the class definition.

Your name: ________________________________

Question 4 (2 points)

The following method reads scores from the keyboard (until the user enters -99) and computes the maximum of the entered scores that are between 0 and 100. The code compiles and prints the correct result, but it violates many coding conventions, good-style practices, and user-friendliness expectations. Briefly describe at least 4 ways this code could be improved.

```
public void PRINT_A_NUMBER()
{
    Scanner zzz = new Scanner(System.in);
    boolean notDone = true;
    int foundIt = -9;

        while (notDone != false)
    {
        System.out.print("Enter score: ");
       int cntScore = zzz.nextInt();

        if (cntScore != -99)
        { if ((cntScore >= 0) && (cntScore <= 100))
            if (cntScore > foundIt)
            foundIt = cntScore; }
         else notDone = false;
     }
    System.out.println(foundIt);
}
```

Your name: ______________________________

Part II – Reading Code

Question 5 (5 points)

a) Consider the following `enigma` method. Fill in the table below that indicates the value returned by `enigma` when the parameter `word` has the value given. Hint: it is similar to your syllable counter, but it does not count syllables (it also does not simply count the total number of vowels in a word). If you carefully trace all variables for 1 or 2 of the sample words, you will quickly discover what it does.

```
public int enigma(String word)
{
    int maxCount = 0, curCount = 0;
    for (int i=0; i < word.length(); i++)
    {
        switch (word.charAt(i))
        {
            case 'a':
            case 'e':
            case 'i':
            case 'o':
            case 'u':
            case 'y':
                curCount++;
                if (curCount > maxCount)
                {
                    maxCount = curCount;
                }
                break;
            default:
                curCount = 0;
        }
    }
    return maxCount;
}
```

Value of parameter `word`	**Value returned by `enigma` method**
serious	
cheese	
yearbook	
queueing	

b) Suggest a better name for the `enigma` method:

Your name: ______________________________

Question 6 (3 points)

a) What does the following code fragment print?

```
int sum = 0;
int i = 0;
while (i < 5)
{
    sum = sum + i;
    i++;
}
System.out.println("i: " + i + ", sum: " + sum);
```

b) The fragment executes more times through the loop than is necessary. Change the fragment to remove the unneeded pass through the loop, but make sure the values of `i` and `sum` that are printed out remain the same.

Question 7 (2 points)

What is the value of d after this segment of code has been executed? Show some work so I know you're not just guessing.

```
boolean a = true, b = false, c = false;
boolean d = ((a && b) || (a && c) || !(b || !c));
```

Question 8 (4 points)

What does the following code fragment print?

```
int x = 0;
for (int i=0; i < 4; i++) {
    for (int j=1; j <= 3; j++) {
        x = i + j;
        System.out.print(x + " ");
    }
    System.out.println();
}
```

Your name: ________________________________

Part III – Find the Bug

Question 9 (2 points)

The printArray method compiles without error, but every time it is called it produces a runtime error. Fix the mistake.

```
public void printArray(int[] data) {

    for(int i = 0; i <= data.length; i++) {

        System.out.println(data[i]);

    }
}
```

Question 10 (2 points)

This code segment should print "probably an infinitive verb" if the String s starts with a lowercase letter and ends with "en", but it doesn't work. Fix the mistake.

```
String s = "spielen";

char firstLetter = s.charAt(0);

String suffix = s.substring(s.length()-2);

boolean startsWithLower = Character.isLowerCase(firstLetter);

if (startsWithLower && (suffix == "en")) {

    System.out.println("probably an infinitive verb");

}
```

Question 11 (2 points)

Consider the familiar `LexUnit` class included at the end of the exam. The code fragment below should print:

```
OrthForms: movie
PartOfSpeech: n
Synonyms:  feature film flick
Definition:  action on continuous film
```

but it prints:

```
OrthForms: n
PartOfSpeech: movie
Synonyms:  feature film flick
Definition:  action on continuous film
```

Find and fix the bug in this code:

```
String line = "movie n 3 feature film flick action on continuous film";

Scanner lineScan = new Scanner(line);

String word = "", pos = "", synonyms = "", definition = "";

int numSynonyms = 0;

LexUnit lex;


word = lineScan.next();

pos = lineScan.next();

numSynonyms = lineScan.nextInt();

lex = new LexUnit(pos, word);


for (int i=0; i < numSynonyms; i++) {

    lex.addSynonym(lineScan.next());

}


lex.setDefinition(lineScan.nextLine());

System.out.println(lex);
```

Your name: ________________________________

Part IV – Writing Code

Question 12 (4 points)

Given the following DrJava interactions, write the class definition for the `Movie` class that works correctly and uses standard coding style conventions (you don't need to write javadoc comments). Each constructor should initialize all instance variables.

```
> Movie m1 = new Movie();
> m1.getDurationInMinutes();
0
> m1.isKidsMovie();
false
> Movie m2 = new Movie(152, true);
> m2.getDurationInMinutes();
152
> m2.isKidsMovie();
true
```

Your name: ________________________________

Question 13 (5 points)

In German, the past participle is usually formed by adding the prefix "ge" and the suffix "t" to the stem of a verb:

gespiel**t** (infinitive: spielen, stem: spiel)

However, if the stem ends with "ier", then the past participle is formed by simply adding the suffix "t" to the stem:

probier**t** (infinitive: probieren, stem: probier)

Fill in the `getPastParticiple` method in the following `GermanVerb` class:

```
public class GermanVerb
{
    private String infinitive;
    private String stem;

    /**
     * Create a GermanVerb with an infinitive form and set its stem
     * by removing the final “en”
     * @param anInfinitive the infinitive form of the verb
     */
    public GermanVerb(String anInfinitive)
    {
        infinitive = anInfinitive;
        stem = infinitive.substring(0, infinitive.length()-2);
    }

    /**
     * Get the past participle of this GermanVerb, which is calculated
     * from its stem
     * @return the past participle of this GermanVerb
     */
    public String getPastParticiple()
    {










    }
}
```

Your name: ________________________________

Bonus (1 point)

In the bonus array below, you can find the hidden message "happy holidays". Fill in the missing control parts of the for-loop so that the code fragment prints:

"happy holidays".

```
char[] bonus = {'e', 's', 'g', 'y', 'a', 'a', 't', 'd', 't',
        'i', 's', 'l', 'e', 'o', 'f', 'h', ' ', ' ', 'e', 'y',
        'h', 'p', 'o', 'p', 'r', 'a', 'f', 'h'};




for (                    ;                          ;             )
{
    System.out.print(bonus[i]);
}
```