

University of Illinois at Urbana-Champaign
 Department of Computer Science
First Examination Fall 2010
 CS 125 Introduction to Computer Science
 90 minutes permitted

First name: _____ Last name: _____

NetID: _ _ _ _ _ @ illinois.edu
(Please write legibly)

Discussion Section: AY _____
 (We will return your exam manuscript to you in this section)

AYA	Tues	9:00am-10:50
AYB	Tues	11:00am-12:50
AYC	Tues	1:00pm- 2:50
AYD	Tues	3:00pm- 4:50
AYE	Wed	9:00am-10:50
AYF	Wed	11:00am-12:50
AYG	Wed	3:00pm- 4:50
AYH	Wed	5:00pm- 6:30

- This is a closed book and closed notes exam. No electronic aids are allowed. You must turn this exam booklet in before leaving.
- You should have 9 sheets total including one scratch sheet. The last sheet is scratch paper; you may detach it while taking the exam, but must turn it in with the exam when you leave.
- You should not write any additional methods for any problem in this exam; you only complete the main() method for that problem. Also, you are not allowed to use the break, continue, or switch statements on this exam.
- Unless we say otherwise in the specific problem, you can assume all values entered by the user will be acceptable input for that program. If you believe a question is ambiguous or contains an error, write down your assumptions and answer accordingly.
- When you write code, you may use a shorthand for System.out and TextIO input and output methods provided it is obvious to the graders which method you are using. For example it is acceptable to use Sopln in place of System.out.println and to use Sopt in place of System.out.print.
- For full marks correct syntax is required: Ensure all statements include a semicolon and the correct use of upper/lower case, single quotes and double quotes. However a syntactically correct but flawed program will earn a low score i.e. semantics and correctness are more important than minor typos and syntax errors.

Problem	Points	Score	Grader
1	12		
2	10		
3	14		
4	15		
5	12		
6	12		
Total	75		

1. Type Analysis – 12 points (3 points each)

For each of the code snippets below, there are **zero**, **one** or **more** type errors that the compiler would catch. If there are one or more types error circle "YES" and explain what the type error(s) are. Indicate what types and operators are involved in the erroneous part of the expression. Circle "No" if there is no type error. Examples:

Example	Response
<code>int a=4+"hi";</code>	Yes: A string value is being assigned to an integer variable.
<code>true==3</code>	Yes: An integer value is being compared with a boolean.
<code>int c=6*3;</code>	No.

(a)

```
double[] data = new double[10];
int result = (data[9] + 4) / 2.0;
```

Circle the correct response (YES or NO). If "YES" then specify the types and the operation involved that cause the type error.

Type Error(s)?	If yes, explain:
YES / NO	

(b)

```
double t = 1 + TextIO.getlnInt();
boolean result = (t != true) || (t > 5);
```

Type Error(s)?	If yes, explain:
YES / NO	

(c)

```
String text = TextIO.getln();
int pos = text.length()/2;
int offset = (text.charAt(pos) - 'A');
```

Type Error(s)?	If yes, explain:
YES / NO	

(d)

```
String text = TextIO.getln();
int mid = text.length()/2;
boolean x = mid > text.indexOf("::");
```

Type Error(s)?	If yes, explain:
YES / NO	

2. Execution Analysis – 10 points

(a) Read the code below then answer the four questions on the right.

```
int a = 1, b = 2, c = 3, d=0;
boolean e = true;
boolean f = false;
while(a<c && !f) {
    c = TextIO.getlnInt();
    d ++;
    e = (2*a == c + b);
    b = c;
    if(d>10 || e) f = true;
}
TextIO.putln(d);
```

What is the smallest value
that can be printed? _____

Choose one variable ('a' to 'f') that best
illustrates the following concepts.

One-way flag _____

Follower/Previous value _____

Stepper/Iteration counter _____

(b)

Add Java code below to print the value of the first element of A multiplied by the last element of A.

```
public static void main(String[] args) {
    int A[] = new int[] {10,20,30, ... more values not shown};

    }
}
```

(c)

Carefully analyze the following code. What will it print when executed? For partial credit show your working.

```
for (int a = 0; a < 5; a++)
    for (int b = a; b > 0; b--)
        TextIO.put(a);
```

Output : _____

(d)

Write one value for x that will cause y to be incremented in the code below.

```
int x = ...not shown...;
int y =0;
if( (x%2 != 0) && (x<0) && (y++ < 10)) TextIO.putln();
// y should now be one.
```

Your answer: x= _____

3. Conditional Cheesy Poofs – 14 Points

Foods4U Corp recommends food based on the user's cravings. Complete the program below to print one of "B", "C", "P", or "S", corresponding to Bacon, Candy, Poofs & Spaghetti respectively, using the following rules:

- If none of the other rules are satisfied, then cheesy-**Poofs** are recommended.
- Cravings for Protein or Salt get **Bacon Strips** if they have more than \$5 to spend.
- Cravings for Carbs get either **Candy** if they have \$1, \$2 or \$3,
or **Spaghetti** if they have more than \$3 to spend.

Read the following carefully: Complete the given code below. Use only if-else statements and **four** output statements – one to print each output. That is, **do not** print out "B" in two different places in the code, or assign it to a variable – your boolean expressions should be such that you do not have multiple statements in your code which print out the same letter. Do not create any additional variables or methods.

```
public class Foods4UWizard {
    public static void main (String[] args){
        TextIO.putln("Do you crave Carbs, Salt or Protein?");
        TextIO.putln("Type C, S, or P and press return");
        char crave = TextIO.getlnChar(); // assume either C, S, or P
        TextIO.putln("Dollars you can spend on this food?");
        int dollars = TextIO.getlnInt(); // assume a valid integer
```

```
// Please continue over-leaf or on spare page if you need more space.
    } // end of main method
} // end of class
```

4. Text Processing (There is no spoon) – 15 Points

Complete the following program to process only the first 64 lines from a text file, which is far longer. Use `TextIO.getln()` to read the next line from the text file. Your program will determine if there are any empty lines (i.e. zero length) *before* a line *starting* with "SPOO", in upper-, lower- or mixed case. There may be other characters on the same line after spoo. Your program will also determine the length of the longest line read. After processing is complete, print the result in the following exact format shown below. Do not print anything else.

```
MAX=81
FOUNDEEMPTYBEFORE=yesno
```

Where *81* is replaced by the length of the longest line and *yesno* is either the word YES or NO. Print "YES" if at least one empty line was found before the "spoo..." line (upper/lower/mixed case eg. SpOo), "NO" in all other cases (e.g. there is no "spoo" line).

The following string methods may be useful: `.equals(string)`, `.length()`, `.charAt(int)`, `.toCharArray()`, `.indexOf(string)`, `.toUpperCase()`.

```
public class SpoonReader {
    public static void main(String[] args) {
        TextIO.readFile("spoon.txt");
```

```
    } // end of main method
} // end of class
```

5. Party Manager (and I know I'm on the invite list) – 12 Points

```
public class PartyInvite {
    public static void main(String[] args) {
        TextIO.putln("Number of people?");
        int len = TextIO.getlnInt();
```

A: String[] names =
boolean[] invited =

```
int total =0;
```

B: for(int i=0;) {

```
    TextIO.putln("Name & Invitation for person "+(i+1)+"?");
    names[i]= TextIO.getln();
    invited[i]= TextIO.getln().equals("Y");
```

C:

D:

```
} // for loop ends here
```

```
TextIO.putln("Total invites:" + total);
TextIO.putln("Do not allow in:");
```

E:

```
} // main
} // class
```

(a) Write code in response area *A* to respectively set *names* and *invited* variables to new String and boolean arrays of length *len*.

(b) Complete the for-loop at response area *B* to read in all *len* people.

Note C and D are inside the for-loop.

(c) Add additional code at *C* so that if the most recently entered name equals your nickname then you are *always* invited (i.e. your boolean entry is true even if user entered "N").

(d). Write code in response area *D* such that the variable *total* will be equal to the total number of invited guests (including yourself).

(e) Write code in response area *E* to print out the names of unwanted (uninvited) people. Print one name per line. Do not print anything else.

6. Cipher (Last Question) – 12 points

You want to decode a message from your secret admirer using a modified Caesar cipher. Your admirer usually mapped every letter of the alphabet to another letter of the alphabet a fixed distance away (i.e., for shift 3 they changed A to D, B to E, Z to C, etc.) but this time they are more cunning. The message **contains only capital letters** A,B,C...Z or **spaces** (there are no other character types). Space always decodes to a space.

Complete the program below: Take a line of encoded text as input from the user (assume it only includes characters A...Z and spaces) and then print the decoded message. The shift is initially 3, thus 'D' should decode to 'A' and 'A' to 'X' etc. *But* each time you come across a space character, increase the shift distance by one. Assume there are less than 22 spaces in the message. Example: if the user enters "FFF FF F" Your program will print "ccc BB A".

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

        TextIO.putln("Please enter the secret (all caps & spaces)");

        String secret = TextIO.getln();

        // Write your code here to decode and print the message
        // String methods .charAt(int) and .length() may be useful.

        } // end of main method
    } // end of class LAST QUESTION. GO BACK AND CHECK YOUR WORK.
```

NetID: _ _ _ _ _

Empty Page

NetID: _ _ _ _ _

Scratch paper - you may detach it while taking the exam, but must turn it in with the exam when you leave.