# CS 1301 – Spring 2009 Exam 3

# Your Name:_____
# Your TA's Name: _____

| Problem | Earned | Possible |
|---|---|---|
| 1. Vocabulary | | 15 |
| 2. Mystery Drawing | | 5 |
| 3. My Picture | | 8 |
| 4. Blast Off | | 9 |
| 5. Break Them Out | | 14 |
| 6. Space The Groups | | 20 |
| 7. Smarter Squaring | | 10 |
| 8. Stock Games | | 10 |
| 9. Know your Sequence | | 9 |
| Extra Credit | | (3) |
| TOTAL: | | 100 (103 w/ ec) |

1. Vocabulary Matching (15 points). Write the number of the correct definition from the right column before each word in the left column.

| | |
|---|---|
| ____ Handle<br>____ Pixel<br>____ Local Variables<br>____ Format Operator<br>____ Immutable<br>____ Short Circuit Evaluation<br>____ Lambda<br>____ Global Variables<br>____ Semantic errors<br>____ File<br>____ Runtime errors<br>____ Exception<br>____ Raise<br>____ Module<br>____ Syntax Errors | 1. The smallest distinct point in a graphic image.<br>2. A named entity, usually stored on a hard drive, floppy disk, or CD-ROM, that contains a stream of characters.<br>3. The `%` operator takes a format string and a tuple of expressions and yields a string that includes the expressions, formatted according to the format string.<br>4. A file containing definitions and statements intended to be imported by other programs.<br>5. A data type in which the elements can not be modified.<br>6. An error that occurs at runtime.<br>7. To prevent an exception from terminating a program using the `try` and `except` statements.<br>8. Statement used to signal an exception.<br>9. Produced by Python when it encounters a problem interpreting code.<br>10. Raised by the runtime system if something goes wrong while the program is running.<br>11. Problems with a program that compiles and runs but doesn't do the right thing. Example: An expression may not be evaluated in the order you expect, yielding an unexpected result.<br>12. Can be seen throughout a program module, even inside of functions.<br>13. Names defined within a function, are only visible within that function.<br>14. A block of code which can be executed as if it were a function but without a name.<br>15. When a boolean expression is evaluated the evaluation starts at the left hand expression and proceeds to the right, stopping when it is no longer necessary to evaluate any further to determine the final outcome. |

2. Read Code: Mystery Drawing (5 Points)

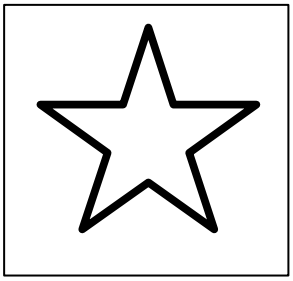
Sketch the output of this code in the provided box (exact precision is not needed, but items should be positioned relatively correctly):

```
from myro import *
win = GraphWin("MyWin", 100, 100)
circle = Circle(Point(50, 50), 45)
circle2 = Circle(Point(30, 35), 15)
circle3 = Circle(Point(70, 35), 15)
line1 = Line(Point(25, 75), Point(75, 75))
line2 = Line(Point(50, 45), Point(40, 65))
line3 = Line(Point(40, 65), Point(60, 65))

circle.draw(win)
circle2.draw(win)
circle3.draw(win)
line1.draw(win)
line2.draw(win)
line3.draw(win)
```

3. Read Code: My Picture (8 points)

In the following code, myPicture.jpg contains the following image and it's dimensions are 200x200 pixels:



Read the following code and sketch what the show() function draws:

```
p = loadPicture("myPicture.jpg")
for x in range( getWidth(p) - 50):
     for y in range( getHeight(p) / 2 ):
          pix = getPixel(p, x, y)
     setRed(pix, 0)
     setGreen(pix, 0)
     setBlue(pix, 0)
show(p)
```

4. Write  Code – BlastOff (9 points)

Write three functions (blastOffWhile, blastOffFor, and blastOffRec). Each function accepts a single parameter which will be a positive integer and prints out a countdown like the following, where N is the number passed into the function:

```
N
.
.
.
3
2
1
Blastoff!
```

a. blastOffWhile must use only a while loop (3 pts):

b. blastOffFor must use only a for loop (3 pts)

c. blastOffRec uses only recursion (no for or while loops!) (3pts):

5. Write Code! (14 points) Break Them Out

You have written some code to collect IR values as follows:

```
aList = []
for x in range(5):
     aList = aList + [ getIR() ]
     DoSomeMovement()
```

This results in aList holding values such as the following:

```
aList = [ [1,0],  [0,0], [1,1], [0,1], [0,0], [1,1] ]
```

But now your professor only wants you to use the RIGHT ir sensor value (the 2nd element in each sublist).

a. (10 pts) Write a function unzip2nd( aList) that returns a list made up of only the 2nd part of each sublist. For example, if you used it on the example aList, it should return a list like this: [0,0,1,1,0,1]. You may NOT use the map function.

b.(4 pts) Now, re-write the unzip2nd() function **without** using a for or while loop. We suggest you use the map function. You may write another function of your choosing to help you out, or use a lambda function.

6. Write Code – Space the Groups (Reading from/ Writing to files) (20 points)

Write a function called spaceGroups( *inFile*, *outFile*) that takes in two string filenames as parameters. The *inFile* will contain a list of names of students, with each line containing one name. However, there's no spacing to this file. You know that every 3 students are in a group, so you want to make a new output file *outFile* that contains all the names, but after every 3rd person, insert a new line to create spaces between the groups. This function should not return anything.

| So if the *inFile* has: | The *outFile* should have: |
| --- | --- |
| Sam<br>Peter<br>Chris<br>Danny<br>Trevor<br>Melody<br>Ami | Sam<br>Peter<br>Chris<br><br>Danny<br>Trevor<br>Melody<br><br>Ami |

7. Write code: (10 points) Smarter Squaring

Write a function called smarterSquaring that takes in no parameters. Your function should prompt the user to enter some input ("Enter a number to square:") using the raw_input function. Your smarterSquaring function must make sure that what the user has entered can be converted to a float (e.g. "thirty point five" would be invalid but "35" and "3.5" would work.) If the input is invalid, the user should be prompted for input again, otherwise the function should return the square of the number that was entered. (For example, if the user typed 4.0, your function should return 16.0). You MUST use a try-except in this function.

8. Computational Complexity: Stock Games (10 points):

You are hired by a BigWinner Inc. to finish their stock recommendation software package after the previous developer was hit by a bus. The previous developer has left you two functions (RateStocksA, and RateStocksB) which are used to predict how much profit stocks will give in the next day. You test out each function with 1 stock, 2 stocks, and 5 stocks and find the following run-time and prediction accuracy results:

| **Number of Stocks** | **RateStocksA** | **RateStocksB** |
|---|---|---|
| 1 | 5 seconds / 89 % accuracy | 1 second / 92% accuracy |
| 2 | 10 seconds / 90 % accuracy | 4 seconds / 92.5 % accuracy |
| 5 | 25 seconds / 89.5 % accuracy | 25 seconds / 91.9 % accuracy |

a. What is the Big O (Computational Complexity) class of each function (N = Number of Stocks) (6 points)?

RateStocksA _____ RateStocksB _____

b. Assuming you want to analyze the top 2500 stocks on the NYSE at the end of one trading day and decide what to purchase by the start of the next day to maximize your profit, which algorithm would you use? WHY? (4 points)

9. Know your sequences! (9 points)

Three of the compound data types you have learned about are sequences. Name these three different types of sequences, give an example of each, and **state why they are different** from each other.

1.

2.

3.

Extra Credit: (3 points)

Write a BlastOffFunc method (as described in the BlastOff test problem) that uses elements of functional programming. It may NOT use a for or while loop, or recursion. You may create a helper function and call map, filter or reduce.