EXAMINATION INSTRUCTIONS

This examination has 9 pages. Check that you have a complete paper. Each candidate should be prepared to produce, upon request, his or her SUNY/UB card. This examination has 8 questions. Answer all questions.

You have 60 minutes to complete this examination. Use your time accordingly.

READ AND OBSERVE THE FOLLOWING RULES:

► Names are pre-printed on the exam booklets. Ensure that you have YOUR exam.
► Sign, using your usual signature, in the space provided on the back cover.
► All of your writing must be handed in. This booklet must not be torn or mutilated in any way, and must not be taken from the examination room.
► Show all of your work in arriving at an answer, unless instructed otherwise. Partial credit will be awarded as appropriate.
► Candidates are not permitted to ask questions of the invigilators, except in cases of supposed errors or ambiguities in examination questions.
► CAUTION – Candidates guilty of any of the following, or similar, dishonest practices shall be immediately dismissed from the examination and shall be liable to disciplinary action.
  ♦ Making use of any books, papers or memoranda, calculators or computers, audio or visual cassette players, or other memory aid devices, other than those explicitly authorized by the examiners.
  ♦ Speaking or communicating with other candidates.
  ♦ Purposely exposing written papers to the view of other candidates. The plea of accident or forgetfulness shall not be received.

---------------------------------- DO NOT WRITE BELOW THIS LINE! ----------------------------------
Module 1 - Question 1  [20 points, 4 points each]
The code given below is syntactically correct.

Circle, and identify by number, one and only one example of each of the following items in the code below. If you believe no example exists, write “no example” next to that item in the list. To show you how I want the question answered, the first one is done for you.

0. a keyword
1. an argument list
2. an assignment operator
3. a function body
4. a parameter list
5. a return statement

```python
def cartTotals( carts, prices ) :
    answer = {}
    for cust in carts.keys() :
        tot = singleCartTotal( carts[cust], prices )
        answer[ cust ] = tot
    return answer
```
Module 1 - Question 2  [20 points total, 10 each part]
[ ] 10 points: perfect  [ ] 3 points: clearly wrong, some correct elements
[ ] 7 points: essentially correct but with small mistakes  [ ] 0 points: for anything else

Study the following code, then answer the question which follows.

```javascript
function whatDoesThisDo(x) {
    let result = "planet";
    if (x < 100) {
        return "star";
    }
    if (x > 200) {
        result = result + "Comet";
    }
    else if (x > 150) {
        result = result + "Asteroid";
    }
    if (x > 175) {
        result = result + "Tail";
    }
    else if (x < 165) {
        result = result + "Field";
    }
    else {
        result = result + "Moon";
    }
    return result;
}
```

[10 points] What does the following statement print?
```javascript
console.log(whatDoesThisDo(167));
```
Write your answer below:

```
planetAsteroidMoon
```

[10 points] Give a value for x so that when whatDoesThisDo(x) is called the function returns "planetField". Write your answer below:
137 (any value from 100 to 150, inclusive, is acceptable)
Module 1 - Question 3  [20 points total]

The following formula defines a parabola:

\[ y = 17x^2 + 2x + 3 \]

Define a Python function `parabola` that accepts an `x` value and returns the corresponding `y` value according to that formula.

Write your answer below:

```python
def parabola(x):
    return 17 * x * x + 2 * x + 3
```
Module 1 - Question 4  [20 points total]

Consider a function named category which will be called with one argument (you can assume that it will be called only with an integer value representing a sensor measurement) that computes a string value representing the corresponding category, according to this table:

<table>
<thead>
<tr>
<th>Sensor measurement</th>
<th>Category</th>
</tr>
</thead>
<tbody>
<tr>
<td>&gt;= 950</td>
<td>Extreme</td>
</tr>
<tr>
<td>&gt;= 750 and &lt; 950</td>
<td>Significant</td>
</tr>
<tr>
<td>&gt;= 532 and &lt; 750</td>
<td>Acceptable</td>
</tr>
<tr>
<td>&gt;= 225 and &lt; 532</td>
<td>Poor</td>
</tr>
<tr>
<td>&lt; 225</td>
<td>Anemic</td>
</tr>
</tbody>
</table>

Give a test case for this function: a possible input along with the correct return value for that input [ 4 points total ]

Input: 715, correct return value: "Acceptable"

Give a definition for the function letterGrade in Python: [ 16 points total ]

```python
def category(gr):
    if gr >= 950: return "Extreme"
    elif gr >= 750: return "Significant"
    elif gr >= 532: return "Acceptable"
    elif gr >= 225: return "Poor"
```
else: return "Anemic"

# Many variations are possible!
Module 2 - Question 1 [20 points, 4 points each]
The code given below is syntactically correct. I have added some extra spacing to make this question easier to answer.

Circle, and identify by number, one and only one example of each of the following items in the code below. If you believe no example exists, write "no example" next to that item in the list. To show you how I want the question answered, the first one is done for you.

0. numeric literal
1. a Boolean expression
2. a keyword
3. the name of a library
4. a for statement (the entire statement)
5. a string literal

import re

def countWords ( filename ) :
    count = {}  

    with open ( filename ) as f :
        for line in f :
            wordList = re.split ( "[^a-zA-Z]+" , line )
            for word in wordList :
                if word in count :
                    count [ word ] = count [ word ] + 1
                else :
                    count [ word ] = 1
        return count

0
1 and 4
The entire statement MUST be identified.
For 4, another possible answer consists of the wordList assignment statement and the entire inner loop.

This is the only dictionary lookup.

count[word] on the LHS of the assignment operator is not a lookup.
Module 2 - Question 2  [20 points total]

Consider a Python function named `sumOfBigs` which has one parameter. The function will be called with a list of integers and will return the sum of all the "big" integers: those that are greater than 50 or less than -50.

Part 1 [1 point]
What value does `sumOfBigs([10, 30, 50, 70])` return?

**70**

Part 2 [1 point]
What value does `sumOfBigs([456, 213, -45, -213, -400])` return?

**56**

Part 3 [1 point]
What value does `sumOfBigs([])` return?

**0**

Part 4 [17 points]

[ ] 17 points: perfect  
[ ] 6 points: clearly wrong, some correct elements  
[ ] 12 points: essentially correct but with small mistakes  
[ ] 0 points: for anything else  

Define, in Python, the function `sumOfEvens`:

```python
def sumOfBigs(x):
    answer = 0
    for v in x:
        if v > 50 or v < -50:
            answer = answer + v
    return answer
```
Module 2 - Question 3  [20 points total]
[ ] 20 points: perfect  
[ ] 7 points: clearly wrong, some correct elements  
[ ] 14 points: essentially correct but with small mistakes  
[ ] 0 points: for anything else

Define a JavaScript function named `createNewArray` with two parameters. Assume the function's first input will be an array of string values (the `substitution`) and the second input will be an array of integer values (the `pattern`). Assume that the integers in the pattern are valid indices for the substitution.

Define the function so it returns a new array containing the string at index `i` from `substitution` if the corresponding entry in `pattern` is `i`. The original array must not be modified.

For example,

```
createNewArray(["sue", "amy", "bob"], [2, 2, 2, 1])
```


Another example:

```
createNewArray(["sue", "amy", "bob"], [0])
```

must return a new array like this:

`["sue"]`

Write your answer below:

```javascript
function createNewArray(substitution, pattern) {
    let result = [ ];
    for (let i of pattern) {
        result.push( substitution[i] );
    }
    return result;
}
```
Module 2 - Question 4  [20 points total]

Study the following code:

```python
import csv

def mystery(filename):
    with open(filename, "r", newline='') as f:
        reader = csv.reader(f)
        for line in reader:
            print(line[3] + " >" + line[0] + line[1] + "<")
        print("Fini")

mystery("f.csv")
```

Give possible contents for the file f.csv which would cause the above to print:

```
d >ab<
4 >12<
@ >$%<
Fini
```

Give your answer below:

*Here's one possible solution. There are many others:*

```
a,b,c,d
1,2,3,4
$,%,,#,@
```