



**ARIZONA STATE UNIVERSITY**  
**GENERAL STUDIES COURSE PROPOSAL COVER FORM**

**Course information:**

Copy and paste **current** course information from [Class Search/Course Catalog](#).

Academic Unit Computing, Informatics, and Decision Systems Engineering Department Computer Science

Subject CSE Number 194 Title Computer Science Principles Units: 3

Is this a cross-listed course? (Choose one)  
 If yes, please identify course(s) \_\_\_\_\_

Is this a shared course? (choose one) If so, list all academic units offering this course \_\_\_\_\_  
 Course description: \_\_\_\_\_

**Requested designation:** Mathematical Studies-CS  
*Note- a **separate** proposal is required for each designation requested*

**Eligibility:**  
 Permanent numbered courses must have completed the university's review and approval process.  
 For the rules governing approval of omnibus courses, contact the General Studies Program Office at (480) 965-0739.

**Area(s) proposed course will serve:**  
 A single course may be proposed for more than one core or awareness area. A course may satisfy a core area requirement and more than one awareness area requirements concurrently, but may not satisfy requirements in two core areas simultaneously, even if approved for those areas. With departmental consent, an approved General Studies course may be counted toward both the General Studies requirement and the major program of study.


- Checklists for general studies designations:**  
 Complete and attach the appropriate checklist
- [Literacy and Critical Inquiry core courses \(L\)](#)
  - [Mathematics core courses \(MA\)](#)
  - [Computer/statistics/quantitative applications core courses \(CS\)](#)
  - [Humanities, Fine Arts and Design core courses \(HU\)](#)
  - [Social and Behavioral Sciences core courses \(SB\)](#)
  - [Natural Sciences core courses \(SO/SG\)](#)
  - [Global Awareness courses \(G\)](#)
  - [Historical Awareness courses \(H\)](#)
  - [Cultural Diversity in the United States courses \(C\)](#)

- A complete proposal should include:**
- Signed General Studies Program Course Proposal Cover Form
  - Criteria Checklist for the area
  - Course Syllabus
  - Table of Contents from the textbook, and/or lists of course materials

**Contact information:**

Name Farideh Tadayon-Navabi, Allison Farina Phone 53228  
 Mail code BYENG 350, BYENG 208 E-mail: navabi@asu.edu  
allison.farina@asu.edu

**Department Chair/Director approval: (Required)**

Chair/Director name (Typed): Ronald Askin Date: 10/10/13  
 Chair/Director (Signature): 

## Arizona State University Criteria Checklist for

### **MATHEMATICAL STUDIES [CS]**

#### **Rationale and Objectives**

The **Mathematical Studies** requirement is intended to ensure that students have skill in basic mathematics, can use mathematical analysis in their chosen fields, and can understand how computers can make mathematical analysis more powerful and efficient. The **Mathematical Studies** requirement is completed by satisfying both the **Mathematics [MA]** requirement and the **Computer/Statistics/Quantitative Applications [CS]** requirement explained below.

The **Mathematics [MA]** requirement, which ensures the acquisition of essential skill in basic mathematics, requires the student to complete a course in College Mathematics, College Algebra, or Precalculus, or demonstrate a higher level of skill by completing a mathematics course for which any of the first three courses in a prerequisite.

The **Computer/Statistics/Quantitative Applications [CS]** requirement, which ensures skill in real world problem solving and analysis, requires the student to complete a course that uses some combination of computers, statistics, and mathematics.

Approved: Feb. 2000

Proposer: Please complete the following section and attach appropriate documentation.

<b>ASU--[CS] CRITERIA</b>			
<b>A COMPUTER/STATISTICS/QUANTITATIVE APPLICATIONS [CS] COURSE MUST SATISFY ONE OF THE FOLLOWING CRITERIA: 1, 2, OR 3</b>			
YES	NO		Identify Documentation Submitted
		<b>1. Computer applications*:</b> courses must satisfy both <b>a</b> and <b>b</b> :	
<input checked="" type="checkbox"/>	<input type="checkbox"/>	<b>a.</b> Course involves the use of computer programming languages or software programs for quantitative analysis, modeling, simulation, animation, or statistics.	See CSE194 Syllabus , using Eclipse to code in Python, learning to program in Python, assignments, textbook table of contents
		<b>b.</b> Course requires students to analyze and implement procedures that are applicable to at least one of the following problem domains ( <b>check those applicable</b> ):	
<input checked="" type="checkbox"/>	<input type="checkbox"/>	<b>i.</b> Spreadsheet analysis, systems analysis and design, and decision support systems.	See CSE194 Syllabus assignemnts 1 through 7, textbook table of contents
<input type="checkbox"/>	<input type="checkbox"/>	<b>ii.</b> Graphic/artistic design using computers.	
<input type="checkbox"/>	<input type="checkbox"/>	<b>iii.</b> Music design using computer software.	
<input type="checkbox"/>	<input type="checkbox"/>	<b>iv.</b> Modeling, making extensive use of computer simulation.	
<input type="checkbox"/>	<input type="checkbox"/>	<b>v.</b> Statistics studies stressing the use of computer software.	
<p>*The <b>computer applications</b> requirement <b>cannot</b> be satisfied by a course, the content of which is restricted primarily to word processing or report preparation skills; learning a computer language or a computer software package; or the study of the social impact of computers. Courses that emphasize the use of a computer software package or the learning of a computer programming language are acceptable, provided that students are required to understand, at an appropriate level, <b>the theoretical principles embodied in the operation of the software and are required to construct, test, and implement procedures that use the software to accomplish tasks in the applicable problem domains.</b></p>			
		<b>2. Statistical applications:</b> courses must satisfy both <b>a</b> and <b>b</b> .	
<input type="checkbox"/>	<input type="checkbox"/>	<b>a.</b> Course has a minimum mathematical prerequisite of College Mathematics, College Algebra, or Precalculus, or a course already approved as satisfying the MA requirement.	

<b>ASU--[CS] CRITERIA</b>			
<input type="checkbox"/>	<input type="checkbox"/>	<b>b.</b> The course must be focused principally on developing knowledge in statistical inference and include coverage of all of the following:	
<b>YES</b>	<b>NO</b>		<b>Identify Documentation Submitted</b>
<input type="checkbox"/>	<input type="checkbox"/>	<b>i.</b> Design of a statistical study.	
<input type="checkbox"/>	<input type="checkbox"/>	<b>ii.</b> Summarization and interpretation of data.	
<input type="checkbox"/>	<input type="checkbox"/>	<b>iii.</b> Methods of sampling.	
<input type="checkbox"/>	<input type="checkbox"/>	<b>iv.</b> Standard probability models.	
<input type="checkbox"/>	<input type="checkbox"/>	<b>v.</b> Statistical estimation	
<input type="checkbox"/>	<input type="checkbox"/>	<b>vi.</b> Hypothesis testing.	
<input type="checkbox"/>	<input type="checkbox"/>	<b>vii.</b> Regression or correlation analysis.	
<b>3. Quantitative applications:</b> courses must satisfy both <b>a</b> and <b>b</b> .			
<input type="checkbox"/>	<input type="checkbox"/>	<b>a.</b> Course has a minimum mathematical prerequisite of College Mathematics, College Algebra, or Precalculus, or a course already approved as satisfying the MA requirement.	
		<b>b.</b> The course must be focused principally on the use of mathematical models in quantitative analysis and design making. Examples of such models are:	
<input type="checkbox"/>	<input type="checkbox"/>	<b>i.</b> Linear programming.	
<input type="checkbox"/>	<input type="checkbox"/>	<b>ii.</b> Goal programming.	

<b>ASU--[CS] CRITERIA</b>			
<input type="checkbox"/>	<input type="checkbox"/>	<b>iii.</b> Integer programming.	
<b>YES</b>	<b>NO</b>		<b>Identify Documentation Submitted</b>
<input type="checkbox"/>	<input type="checkbox"/>	<b>iv.</b> Inventory models.	
<input type="checkbox"/>	<input type="checkbox"/>	<b>v.</b> Decision theory.	
<input type="checkbox"/>	<input type="checkbox"/>	<b>vi.</b> Simulation and Monte Carlo methods.	
<input type="checkbox"/>	<input type="checkbox"/>	<b>vii.</b> Other (explanation must be attached)	

Course Prefix	Number	Title	Designation
CSE	194	Computer Science Principles	CS

Explain in detail which student activities correspond to the **specific** designation criteria.  
Please use the following organizer to explain how the criteria are being met.

Criteria (from checksheet)	How course meets spirit (contextualize specific examples in next column)	Please provide detailed evidence of how course meets criteria (i.e., where in syllabus)
Course uses Python programming language to solve computational problem through the process of design, implementation, documentation, and testing.	<p>The primary goal of this course is to teach problem solving using the Python programming language. The course introduces students to the following topics: Elementary Programming Data &amp; Variables in Python Strings Control structures: if-else &amp; loops Functions Lists, Classes &amp; Objects Recursion</p> <p>The course also gives students an understanding of the breadth of Computer Science as a discipline and how it exists in the world.</p>	<p>There are 8 programming assignments that covers the programming concepts, which is 40% of their grade, 8 quizzes which is 10% of the grade, and 3 exams 50% of the grade. Please see the attached zip file which includes assignments and exams. Please see the brief contents for required book in Python.</p> <p>The slides covers applications of computer science in society by giving an introduction to the following topics: Software Engineering, blogging, database, and gaming. Students are tested on the exam. Also see the table of contents attached for the required book "Computer Science An Overview.</p>

# Computer Science Principles

## CSE 194 – Spring2013

**Course Overview:** This course is designed to give students a broad introduction to the exciting field of computer science.

**Instructor:** Faye Navabi, office located in the Brickyard room 518, Phone: 480-965-3228  
Email: [navabi@asu.edu](mailto:navabi@asu.edu)

### **Course Objectives and Expected Learning Outcomes**

1. To give students the tools to take a computational problem through the process of design, implementation, documentation, and testing.

Objectives:

- Break a broad problem down into specific sub-problems
- Write an algorithm to solve a specific problem, and then translate that algorithm into a program in a specific programming language (Python)
- Write clear, concise documentation
- Develop test cases that reveal programming bugs

2. To give students an understanding of the breadth of Computer Science as a discipline and how it exists in the world.

Objectives:

- Identify applications of computer science in society
- Describe the big questions in computer science
- Describe the relationship between a number of major sub-disciplines within computer science, including functional and imperative programming, computer architecture, and theoretical computer science

**Grading Policies:** There will be regular homework assignments (40% of grade), regular quizzes (10% of grade), 2 in-class midterms and 1 final exam (50% of grade).

**Exams:** There will be two midterm exams and one final exam. The exams are in class and are closed book. However, you are allowed to bring one double-sided sheet of notes (hand-written or typed) to each midterm exams and final exam.

All students must have a passing grade in both the exam component (midterm and final) of the class and the homework component of the class in order to pass the class.

## Grade Breakdown

Final Grade	Percentage
A+	$\geq 97\%$
A	$\geq 90\%$ and $< 97\%$
B+	$\geq 87\%$ and $< 90\%$
B	$\geq 80\%$ and $< 87\%$
C+	$\geq 77$ and $< 80\%$
C	$\geq 70\%$ and $< 77\%$
D	$\geq 60\%$ and $< 70\%$
E	$< 60\%$

**Attendance:** Attending class is important in order for you to be aware of what is going on. Often, announcements will be made or information will be discussed that is not available on the web site.

### Required textbook:

- Computer Science An overview by J. Glenn Brookshear
- Introduction to Programming Using Python by Y. Daniel Liang

### Classroom Behavior

Students in this class are expected to treat others fairly, with respect and courtesy, regardless of such factors as race, religion, sexual orientation, gender, disability, age, or national origin. In this class, you are expected to contribute to the overall campus climate such that others feel welcome, are respected, and are able to develop to their full potential. This will allow each person to contribute to the success of the class as a whole. ASU and the College of Engineering are committed to maintaining a productive, enjoyable and diverse campus environment.

Students are expected to effectively communicate ideas. Inappropriate language (written and oral) does not effectively communicate your ideas to an audience. Inappropriate language includes not only profanity, but also words that are demeaning to a person or group (racially, sexually, ethnically, etc.).

You are expected to participate in the various classroom activities, including:

- coming to each class on time and staying until dismissed;
- following instructions given by the instructor, including actively working on whatever assignment has been given;
- not consuming any food or drink while in the ASU classrooms, and not bringing any open containers of food or drink into the classrooms; and
- avoiding disruptive side conversations.

**Academic Integrity:** All students in this class are subject to ASU's Academic Integrity Policy (available at <http://provost.asu.edu/academicintegrity>) and should acquaint themselves with its content and requirements, including a strict prohibition against plagiarism. All violations will be reported to the Dean's office, who maintains records of all offenses.



There are a number of actions that constitute a violation of the ASU's Academic Integrity Policy. These actions **include, but are not limited to:**

- practicing any form of academic deceit;
- referring to materials or sources or employing devices (e.g., audio recorders, crib sheets, calculators, solution manuals, or commercial research services) not specifically authorized by the instructor for use during tests, quizzes, homework, and class activities;
- acting as a substitute for another person in any academic evaluation or using a substitute in any academic evaluation;
- possessing, buying, selling, or otherwise obtaining or using, without appropriate authorization, a copy of any materials intended to be used for academic evaluation in advance of its administration;
- depending on the aid of others to the extent that the work is not representative of the student's abilities, knowing or having good reason to believe that this aid is not authorized by the instructor;
- providing inappropriate aid to another person, knowing or having good reason to believe the aid is not authorized by the instructor;
- submitting the ideas or work of another person or persons without customary and proper acknowledgment of sources (i.e., engaging in plagiarism);
- permitting one's own ideas or work to be submitted by another person without the instructor's authorization; or
- attempting to influence or change any academic evaluation or record for reasons having no relevance to class achievement.

University policy allows for cheating sanctions ranging from zero credit for an assignment to expulsion (without expectation of readmission) from the University. **Any student who is found to have violated the University's Academic Integrity Policy in this course, no matter how minor the violation, will at a minimum receive an E in the course.**

**Disability Accommodations:** Suitable accommodations will be made for students having disabilities and students should notify the instructor as early as possible if they will require same. Such students must be registered with the Disability Resource Center and provide documentation to that effect.

**Notice:** Any information in this syllabus may be subject to change with reasonable advance notice.

## Tentative Schedule for CSE194 - Spring 2013

This is a tentative schedule and subject to change. Please come to the class and/or check the announcements on the course web page for any changes. Announcements in class take precedence over printed materials

Week (Tue. & Thur.)	Tuesday	Thursday	Assignments Due Dates
1. Jan 8 & 10	Introduction & History of Computing	Data Storage & Number Systems	
2. Jan 15 & 17	Elementary Programming Data & Variables in Python	Mathematical functions , Strings & Objects	
3. Jan 22 & 24	Booleans & Selections	Continue with Selection	<b>Assignment 1 (intro) due Jan 24th</b>
4. Jan 29 & 31	Loops	Loops Continue	
5. Feb 5 & 7	Functions	More on Functions	<b>Assignment 2 (control flow- if-else) due on Feb 14</b>
6. Feb 12 & 14	More on Strings	Introduction to Blogging	
7. Feb 19 & 21	List	List Continue	<b>Assignment 3 (control flow-loops) (due on Feb 21)</b>
8. Feb 26 & 28	Introduction to Software Engineering	Midterm Exam 1	
9. Mar 5 & 7	Introduction to Gaming	Robotics	<b>Assignment 4 (functions) due on Mar 7</b>
10. Mar 12 & 14	Spring Break (Enjoy!!)	Spring Break	
11. Mar 19 & 21	Google Scale Data Management	Object-Oriented Programming	<b>Assignment 5 (class) due on Mar 21</b>
12. Mar 26 & 28	Object-Oriented Programming continue	Class Abstraction & Encapsulation	<b>Course Withdrawal Deadline March 31</b>
13. April 2 & 4	Object-oriented Thinking	Introduction to Security	<b>Assignment 6 (advance class) due on April 4</b>
14. Apr 9 & 11	Review	Midterm Exam 2	
15. Apr 16 & 18	Recursion	Continue Recursion	

16. Apr 23 & 25	More on Designing Classes	Review	<b>Assignment 7 (List) due on April 25</b>
Apr 30	Final Review		



# contents

## **Chapter 0 Introduction 1**

- 0.1 The Role of Algorithms 2
- 0.2 The History of Computing 4
- 0.3 The Science of Algorithms 10
- 0.4 Abstraction 11
- 0.5 An Outline of Our Study 12
- 0.6 Social Repercussions 13

## **Chapter 1 Data Storage 19**

- 1.1 Bits and Their Storage 20
- 1.2 Main Memory 26
- 1.3 Mass Storage 29
- 1.4 Representing Information as Bit Patterns 35
- \*1.5 The Binary System 42
- \*1.6 Storing Integers 47
- \*1.7 Storing Fractions 53
- \*1.8 Data Compression 58
- \*1.9 Communication Errors 63

## **Chapter 2 Data Manipulation 73**

- 2.1 Computer Architecture 74
- 2.2 Machine Language 77
- 2.3 Program Execution 83
- \*2.4 Arithmetic/Logic Instructions 90
- \*2.5 Communicating with Other Devices 94
- \*2.6 Other Architectures 100

*\*Asterisks indicate suggestions for optional sections.*

**Chapter 3 Operating Systems 109**

- 3.1 The History of Operating Systems 110
- 3.2 Operating System Architecture 114
- 3.3 Coordinating the Machine's Activities 122
- \*3.4 Handling Competition Among Processes 125
- 3.5 Security 130

**Chapter 4 Networking and the Internet 139**

- 4.1 Network Fundamentals 140
- 4.2 The Internet 149
- 4.3 The World Wide Web 158
- \*4.4 Internet Protocols 167
- 4.5 Security 173

**Chapter 5 Algorithms 187**

- 5.1 The Concept of an Algorithm 188
- 5.2 Algorithm Representation 191
- 5.3 Algorithm Discovery 198
- 5.4 Iterative Structures 204
- 5.5 Recursive Structures 214
- 5.6 Efficiency and Correctness 222

**Chapter 6 Programming Languages 239**

- 6.1 Historical Perspective 240
- 6.2 Traditional Programming Concepts 248
- 6.3 Procedural Units 260
- 6.4 Language Implementation 268
- 6.5 Object-Oriented Programming 276
- \*6.6 Programming Concurrent Activities 283
- \*6.7 Declarative Programming 286

**Chapter 7 Software Engineering 299**

- 7.1 The Software Engineering Discipline 300
- 7.2 The Software Life Cycle 302
- 7.3 Software Engineering Methodologies 306
- 7.4 Modularity 308
- 7.5 Tools of the Trade 316
- 7.6 Quality Assurance 324
- 7.7 Documentation 328
- 7.8 The Human-Machine Interface 329
- 7.9 Software Ownership and Liability 332

**Chapter 8 Data Abstractions 341**

- 8.1 Basic Data Structures 342
- 8.2 Related Concepts 345
- 8.3 Implementing Data Structures 348
- 8.4 A Short Case Study 362
- 8.5 Customized Data Types 367
- \*8.6 Classes and Objects 371
- \*8.7 Pointers in Machine Language 372

**Chapter 9 Database Systems 383**

- 9.1 Database Fundamentals 384
- 9.2 The Relational Model 389
- \*9.3 Object-Oriented Databases 400
- \*9.4 Maintaining Database Integrity 402
- \*9.5 Traditional File Structures 406
- 9.6 Data Mining 414
- 9.7 Social Impact of Database Technology 416

**Chapter 10 Computer Graphics 425**

- 10.1 The Scope of Computer Graphics 426
- 10.2 Overview of 3D Graphics 428
- 10.3 Modeling 430
- 10.4 Rendering 439
- \*10.5 Dealing with Global Lighting 449
- 10.6 Animation 452

**Chapter 11 Artificial Intelligence 461**

- 11.1 Intelligence and Machines 462
- 11.2 Perception 467
- 11.3 Reasoning 473
- 11.4 Additional Areas of Research 484
- 11.5 Artificial Neural Networks 489
- 11.6 Robotics 497
- 11.7 Considering the Consequences 500

**Chapter 12 Theory of Computation 509**

- 12.1 Functions and Their Computation 510
- 12.2 Turing Machines 512
- 12.3 Universal Programming Languages 516
- 12.4 A Noncomputable Function 522
- 12.5 Complexity of Problems 527
- \*12.6 Public-Key Cryptography 536

# BRIEF CONTENTS

1	Introduction to Computers, Programs, and Python	1	Chapters 16–23 are bonus Web chapters
2	Elementary Programming	31	16 Developing Efficient Algorithms
3	Mathematical Functions, Strings, and Objects	63	17 Sorting
4	Selections	91	18 Linked Lists, Stacks, Queues, and Priority Queues
5	Loops	133	19 Binary Search Trees
6	Functions	171	20 AVL Trees
7	Objects and Classes	215	21 Hashing: Implementing Dictionaries and Sets
8	More on Strings and Special Methods	241	22 Graphs and Applications
9	GUI Programming Using Tkinter	271	23 Weighted Graphs and Applications
10	Lists	313	APPENDIXES
11	Multidimensional Lists	361	A Python Keywords
12	Inheritance and Polymorphism	399	B The ASCII Character Set
13	Files and Exception Handling	439	C Number Systems
14	Tuples, Sets, and Dictionaries	475	INDEX
15	Recursion	499	CREDITS

# CSE 194 - Exam 1 – Spring 2013

Last Name (print) \_\_\_\_\_ First Name \_\_\_\_\_

**MULTIPLE CHOICE (2 points each, choose the ONE correct answer, and write the letter in the space provided)**

\_\_\_\_\_ 1. Why do computers use zeros and ones?

- A. because combinations of zeros and ones can represent any numbers and characters.
- B. because digital devices have two stable states and it is natural to use one state for 0 and the other for 1.
- C. because binary numbers are simplest.
- D. because binary numbers are the bases upon which all other number systems are built.

\_\_\_\_\_ 2. One byte has \_\_\_\_\_ bits.

- A. 4
- B. 8
- C. 12
- D. 16

\_\_\_\_\_ 3. \_\_\_\_\_ is an interpreted programming language.

- A. Python
- B. C++
- C. C
- D. Ada

\_\_\_\_\_ 4. Python syntax is case-sensitive.

- A. True
- B. False

\_\_\_\_\_ 5. Which of the following statement prints smith\exam1\test.txt?

- A. `print("smith\exam1\test.txt")`
- B. `print("smith\\exam1\\test.txt")`
- C. `print("smith\"exam1\"test.txt")`
- D. `print("smith"\exam1"\test.txt")`

\_\_\_\_\_ 6. A Python line comment begins with \_\_\_\_\_.

- A. //
- B. /\*
- C. #
- D. \$\$

\_\_\_\_\_ 7. What is the result of `eval("1 + 3 * 2")`?

- A. "1 + 3 \* 2"
- B. 7
- C. 8
- D. "1 + 6"



\_\_\_\_\_ 8. What will be displayed by the following code?

```
x = 1
x = 2 * x + 1
print(x)
```

- A. 0
- B. 1
- C. 2
- D. 3
- E. 4

\_\_\_\_\_ 9. What will be displayed by the following code?

```
x = 1
x = x + 2.5
print(x)
```

- A. 1
- B. 2
- C. 3
- D. 3.5
- E. The statements are illegal

\_\_\_\_\_ 10. Which of the following expression results in a value 1?

- A.  $2 \% 1$
- B.  $15 \% 4$
- C.  $25 \% 5$
- D.  $37 \% 6$

\_\_\_\_\_ 11. What is the result of evaluating  $2 + 2 ** 3 / 2$ ?

- A. 4
- B. 6
- C. 4.0
- D. 6.0

\_\_\_\_\_ 12. What is x after the following statements?

```
x = 1
x *= x + 1
```

- A. x is 1
- B. x is 2
- C. x is 3
- D. x is 4

\_\_\_\_\_ 13. Which of the following statements are the same?

- (A)  $x -= x + 4$
- (B)  $x = x + 4 - x$
- (C)  $x = x - (x + 4)$

- A. (A) and (B) are the same
- B. (A) and (C) are the same
- C. (B) and (C) are the same
- D. (A), (B), and (C) are the same

\_\_\_\_\_ 14. To add number to sum, you write (Note: Python is case-sensitive)

- A. number += sum
- B. number = sum + number
- C. sum = Number + sum
- D. sum += number
- E. sum = sum + number

\_\_\_\_\_ 15. Is pow(a, b) the same as a \*\* b?

- A. Yes
- B. No

\_\_\_\_\_ 16. Suppose x is 345.3546, what is format(x, "10.3f")? (note b represents a blank space)

- A. bb345.355
- B. bbb345.355
- C. bbbb345.355
- D. bbb345.354
- E. bbbb345.354

\_\_\_\_\_ 17. The equal comparison operator is \_\_\_\_\_.

- A. <>
- B. !=
- C. ==
- D. =

\_\_\_\_\_ 18. The "less than or equal to" comparison operator is \_\_\_\_\_.

- A. <
- B. <=
- C. =<
- D. <<
- E. !=

\_\_\_\_\_ 19. The following code displays \_\_\_\_\_.

```
temperature = 50
if temperature >= 100:
    print("too hot")
elif temperature <= 40:
    print("too cold")
else:
    print("just right")
```

- A. too hot
- B. too cold
- C. just right
- D. too hot too cold just right

\_\_\_\_\_ 20. Assume x = 4 and y = 5, which of the following is true?

- A. not (x == 4)
- B. x != 4
- C. x == 5
- D. x != 5

**21.** Show the screen output generated by the following program (line for line)?  
Hint: true and false are valid results (14 pts)

```
j = 7
k = 3
x = 2.2
print j / k      #1. _____
print j % k      #2. _____

j = j + 2;

print ("j is now: " , j) #3. _____

myGrade = 'A'

print("letter is: " , myGrade) #4. _____

print 5 / 2.0      #5. _____

print 1+ 2 - 3 * 4 / 5 #6. _____

print j != j      #7. _____
```

**22.** Show the exact output of the following code: (5 pts)

```
myList = ["Python", "Java", "C++"]
print(myList[0])
print(myList[-2])
print myList[0:3]
myList.append("Assembly")
for element in myList:
    print element
```

**23.** Show the exact output: (7 pts)

```
def main():
    times = 4 # initialize times
    #Invoke the function nPrint
    nPrint("Welcome to CSE194!" , times)

def nPrint(message, n):
    while n>0:
        print"n= " , n
        print(message)
        n -= 1

main()
```

24. Write the output generated by each code segment given these initializations of j and x: (6 pts)

```
j = 8
x = -1.5
a.  if( x < -1.0 ):
      print( "true" )
      else:
          print( "false" )

b.  if( x >= j ):
      print( "x is high" )
      else:
          print( "x is low" )

c.  if( x <= 0.0 ):
      if( x < 0.0 ):
          print( "neg" )
      else:
          print( "zero" )
      else:
          print( "pos" )
```

25. Show the output: (8 pts)

```
s = 'Arizona State University'
print s[0] _____

print s[2] _____

print s[len(s)-1] _____

print s[8:16] _____
```

26. Given the following function (4 pts)

```
def nPrint(message, n):
    while n > 0:
        print(message)
        n -= 1
```

What will be displayed by the call `nPrint('a', 4)`?

27. What will be displayed by the following code? (4 pts)

```
x = 1
def f1():
    y = x + 2
    print(y)

f1()
print(x)
```

**28.** What is sum after the following loop terminates? (3 pts)

```
sum = 0
item = 0
while item < 5:
    item += 1
    sum += item
    if sum > 4: break

print(sum)
```

**29.** What is the output for y? (3 pts)

```
y = 0
for i in range(0, 10):
    y += i

print(y)
```

**30.** Write a function definition called max that returns the maximum of two integers. For example calling max(5, 4) returns 5. (6 pts)

```
def max(num1, num2):
    # complete the code here
```

## CSE 194 - Exam 2 – Spring 2013

Last Name (print) \_\_\_\_\_ First Name \_\_\_\_\_ Date: \_\_\_\_\_

**MULTIPLE CHOICE (2 points each, choose the ONE correct answer, and write the letter in the space provided)**

\_\_\_\_\_ 1. Given the following function

```
def nPrint(message, n):  
    while n > 0:  
        print(message)  
    n -= 1
```

What will be displayed by the call `nPrint('a', 4)`?

- A. aaaaa
- B. aaaa
- C. aaa
- D. invalid call
- E. infinite loop

\_\_\_\_\_ 2. Given the following function

```
def nPrint(message, n):  
    while n > 0:  
        print(message)  
    n -= 1
```

What is `k` after invoking `nPrint("A message", k)`?

```
k = 2  
nPrint("A message", k)
```

- A. 0
- B. 1
- C. 2
- D. 3

\_\_\_\_\_ 3. A variable defined inside a function is referred to as \_\_\_\_\_.

- A. a global variable
- B. a function variable
- C. a block variable
- D. a local variable

\_\_\_\_\_ 4. A variable defined outside a function is referred to as \_\_\_\_\_.

- A. a global variable
- B. a function variable
- C. a block variable
- D. a local variable

\_\_\_\_\_ 5. Python syntax is case-sensitive.

- A. True
- B. False

\_\_\_\_\_ 6. What will be displayed by the following code?

```
x = 1
def f1():
    y = x + 2
    print(y)

f1()
print(x)
```

- A. 1 3
- B. 3 1
- C. The program has a runtime error because x is not defined.
- D. 1 1
- E. 3 3

\_\_\_\_\_ 7. What will be displayed by the following code?

```
def f1(x = 1, y = 2):
    x = x + y
    y += 1
    print(x, y)

f1()
```

- A. 1 3
- B. 3 1
- C. The program has a runtime error because x and y are not defined.
- D. 1 1
- E. 3 3

\_\_\_\_\_ 8. What will be displayed by the following code?

```
x = 1
x = 2 * x + 1
print(x)
```

- A. 0
- B. 1
- C. 2
- D. 3
- E. 4

\_\_\_\_\_ 9. If a function does not return a value, by default, it returns \_\_\_\_\_.

- A. None
- B. int
- C. double
- D. public

\_\_\_\_\_ 10. The header of a function consists of \_\_\_\_\_.

- A. function name
- B. parameter list
- C. function name and parameter list

\_\_\_\_\_ 11. A function \_\_\_\_\_.

- A. must have at least one parameter
- B. may have no parameters
- C. must always have a return statement to return a value
- D. must always have a return statement to return multiple values

\_\_\_\_\_ 12. Arguments to functions always appear within \_\_\_\_\_.

- A. brackets
- B. curly braces
- C. quotation marks
- D. parentheses

\_\_\_\_\_ 13. \_\_\_\_\_ is a template, blueprint, or contract that defines objects of the same type.

- A. A class
- B. An object
- C. A method
- D. A data field

\_\_\_\_\_ 14. An object is an instance of a \_\_\_\_\_.

- A. program
- B. class
- C. method
- D. data

\_\_\_\_\_ 15. The keyword \_\_\_\_\_ is required to define a class.

- A. def
- B. return
- C. class
- D. All of the above.

\_\_\_\_\_ 16. \_\_\_\_\_ is used to create an object.

- A. A constructor
- B. A class
- C. A value-returning method
- D. A None method



\_\_\_\_\_ 17. Analyze the following code:

```
class A:
    def __init__(self, s):
        self.s = s

    def print(self):
        print(s)

a = A("Welcome")
a.print()
```

- A. The program has an error because class A does not have a constructor.
- B. The program has an error because class A should have a print method with signature print(self, s).
- C. The program has an error because class A should have a print method with signature print(s).
- D. The program would run if you change print(s) to print(self.s).

\_\_\_\_\_ 18. Analyze the following code:

```
class A:
    def __init__(self, s):
        self.s = s

    def print(self):
        print(self.s)

a = A()
a.print()
```

- A. The program has an error because class A does not have a constructor.
- B. The program has an error because s is not defined in print(s).
- C. The program runs fine and prints nothing.
- D. The program has an error because the constructor is invoked without an argument.

\_\_\_\_\_ 19. Given the declaration `x = Circle()`, which of the following statement is most accurate.

- A. x contains an int value.
- B. x contains an object of the Circle type.
- C. x contains a reference to a Circle object.
- D. You can assign an int value to x.

\_\_\_\_\_ 20. Suppose s is "Welcome", what is s.upper()?

- A. welcome
- B. WELCOME
- C. Welcome

\_\_\_\_\_ 21. A(n) method is automatically called when an object is created.

- A. `__init__`
- B. `init`
- C. `__str__`
- D. `__object__`

\_\_\_\_\_ 22. Object reusability has been a factor in the increased use of object-oriented programming?

- A. true
- B. false

23. Show the exact output: (5 pts)

```
def main():
    myCount = Count()
    times = 0

    for i in range(0, 100):
        increment(myCount, times)

    print("myCount.count =", myCount.count, "times =", times)

def increment(c, times):
    c.count += 1
    times += 1

class Count:
    def __init__(self):
        self.count = 0

main()
```

24. Show the exact output: (5 pts)

```
class Name:
    def __init__(self, firstName, mi, lastName):
        self.firstName = firstName
        self.mi = mi
        self.lastName = lastName

firstName = "John"
name = Name(firstName, 'F', "Smith")
firstName = "Peter"
name.lastName = "Pan"
print(name.firstName, name.lastName)
```

25. What are the results of the following expression? (8 pts)

1. `len("Good")` \_\_\_\_\_
2. `max("Programming is fun")` \_\_\_\_\_
3. `s = "Welcome" what is s[2:5]` \_\_\_\_\_
4. `s = "Welcome" what is s.count('e')` \_\_\_\_\_
5. `s = "Programming is fun", what is s.find('ram')` \_\_\_\_\_
6. `s = "Programming is fun", what is s.find('m')` \_\_\_\_\_
7. `s = "Programming is fun", what is s.endswith('fun')` \_\_\_\_\_
8. `s = "Programming is fun", what is s.endswith('n')` \_\_\_\_\_

26. Show the output: (8 pts)

```
s = 'Arizona State University'  
print s[0] _____  
print s[2] _____  
print s[len(s)-1] _____  
print s[8:16] _____
```

(8 pts)

27. Write a function called `sum` that takes two integers and returns the sum of the values between the two integers; For example `sum(1, 10)` returns 55 and `sum(20, 37)` returns 513.

(24 pts)

- **28.** Design a class named **BankAccount** to represent a **bank account** object. The class contains: A private string data field named **id** that keeps track of account's id, and a private float data field named **balance** for the account balance.

**Implement the following methods:**

- A constructor, **init**, that creates a bank account object with the specified id and balance that are passed as arguments to the init method. (Hint: The init method that takes three arguments; self, balance and id.
- A method named **deposit** that adds the specified amount to the balance
- A method named **withdraw** that withdraws a specified amount from the balance. It checks for sufficient funds. Returns false if there are insufficient funds in the account.
- A method named **get Balance()** that return the balance for each account
- Draw a **UML diagram** for the class Bank Account, and then implement the class.

**Here is a sample of using the BankAccount class:**

```
>>> my_account = BankAccount("2345", 200)
>>> my_account.getBalance()
200
>>> my_account.withdraw(200)
True
>>> my_account.getBalance()
0
>>> my_account.withdraw(200)
False
>>> my_account.deposit(450)
>>> my_account.withdraw(250)
True
>>> my_account.getBalance()
200
```

# CSE 194 – Final Exam – Spring 2013

Last Name (print) \_\_\_\_\_ First Name \_\_\_\_\_ Date: \_\_\_\_\_

**MULTIPLE CHOICE (2 points each, choose the ONE correct answer, and write the letter in the space provided)**

\_\_\_\_\_ 1. Suppose list1 is [3, 4, 5, 20, 5, 25, 1, 3], what is len(list1)?

- A. 6
- B. 7
- C. 8
- D. 5
- E. 4

\_\_\_\_\_ 2. Suppose list1 is [3, 4, 5, 20, 5, 25, 1, 3], what is max(list1)?

- A. 5
- B. 4
- C. 8
- D. 25
- E. 1

\_\_\_\_\_ 3. Suppose list1 is [3, 4, 5, 20, 5, 25, 1, 3], what is min(list1)?

- A. 5
- B. 4
- C. 8
- D. 25
- E. 1

\_\_\_\_\_ 4. Suppose list1 is [1, 3, 2], what is sum(list1)?

- A. 5
- B. 4
- C. 6
- D. 2
- E. 1

\_\_\_\_\_ 5. Suppose list1 is [1, 3, 2, 4, 5, 2, 1, 0], What is list1[-1]?

- A. 3
- B. 5
- C. 1
- D. 0

\_\_\_\_\_ 6. Suppose list1 is [1, 3, 2, 4, 5, 2, 1, 0], What is list1[:-1]?

- A. 0
- B. [1, 3, 2, 4, 5, 2, 1]
- C. [1, 3, 2, 4, 5, 2]
- D. [1, 3, 2, 4, 5, 2, 1, 0]

\_\_\_\_\_ 7. Suppose list1 is [1, 3, 2], What is list1 \* 2?

- A. [2, 6, 4]
- B. [1, 3, 2, 1, 3]
- C. [1, 3, 2, 1, 3, 2]
- D. [1, 3, 2, 3, 2, 1]

\_\_\_\_\_ 8. What will be displayed by the following code?

```
myList = [1, 5, 7, 5, 5, 4]
max = myList[0]
for i in range(1, len(myList)):
    if myList[i] > max:
        max = myList[i]

print(max)
```

- A. 0
- B. 1
- C. 3
- D. 5
- E. 7

\_\_\_\_\_ 9. What will be displayed by the following code?

```
myList = [1, 2, 3, 4, 5, 6]
for i in range(1, 6):
    myList[i - 1] = myList[i]

for i in range(0, 6):
    print(myList[i], end = " ")
```

- A. 2 3 4 5 6 1
- B. 6 1 2 3 4 5
- C. 2 3 4 5 6 6
- D. 1 1 2 3 4 5
- E. 2 3 4 5 6 1

\_\_\_\_\_ 10. What will be displayed by the following code?

```
x = 1
x = 2 * x + 2
print(x)
```

- A. 0
- B. 1
- C. 2
- D. 3
- E. 4

\_\_\_\_\_ 11. What will be displayed by the following code?

```
list1 = [1, 3]
list2 = list1
list1[0] = 4
print(list2)
```

- A. [1, 3]
- B. [4, 3]
- C. [1, 4]
- D. [1, 3, 4]

\_\_\_\_\_ 12. What is `chr(ord('B'))`?

- A. A
- B. B
- C. C
- D. D

\_\_\_\_\_ 13. Suppose `x` is a char variable with a value 'b'. What will be displayed by the statement `print(chr(ord(x) + 1))`?

- A. a
- B. b
- C. c
- D. d

\_\_\_\_\_ 14. \_\_\_\_\_ is a template, blueprint, or contract that defines objects of the same type.

- A. A class
- B. An object
- C. A method
- D. A data field

\_\_\_\_\_ 15. An object is an instance of a \_\_\_\_\_.

- A. program
- B. class
- C. method
- D. data

\_\_\_\_\_ 16. The keyword \_\_\_\_\_ is required to define a new class.

- A. def
- B. return
- C. class
- D. All of the above.

\_\_\_\_\_ 17. What is the type for object 5.6?

- A. int
- B. float
- C. str

\_\_\_\_\_ 18. Analyze the following code:

```
class A:
    def __init__(self, s):
        self.s = s

    def print(self):
        print(self.s)

a = A()
a.print()
```

- A. The program has an error because class A does not have a constructor.
- B. The program has an error because s is not defined in print(s).
- C. The program runs fine and prints nothing.
- D. The program has an error because the constructor is invoked without an argument.

\_\_\_\_\_ 19. Given the declaration `x = Circle()`, which of the following statement is most accurate.

- A. x contains an int value.
- B. x contains an object of the Circle type.
- C. x contains a reference to a Circle object.
- D. You can assign an int value to x.

\_\_\_\_\_ 20. Which of the following code displays the area of a circle if the radius is positive.

- A. `if radius != 0: print(radius * radius * 4.14159)`
- B. `if radius >= 0: print(radius * radius * 4.14159)`
- C. `if radius > 0: print(radius * radius * 4.14159)`
- D. `if radius <= 0: print(radius * radius * 4.14159)`

\_\_\_\_\_ 21. What is the output of the following code?

```
x = 0
if x < 4:
    x = x + 1

print("x is", x)
```

- A. x is 0
- B. x is 1
- C. x is 2
- D. x is 4
- E. x is 6

\_\_\_\_\_ 22. How many times will the following code print "Welcome to Python"?

```
count = 1
while count < 10:
    print("Welcome to Python")
    count += 1
```

- A. 8
- B. 9
- C. 10
- D. 11
- E. 0



\_\_\_\_\_ 23. What is the output of the following code?

```
x = 0
while x < 4:
    x = x + 1

print("x is", x)
```

- A. x is 0
- B. x is 1
- C. x is 2
- D. x is 3
- E. x is 4

\_\_\_\_\_ 24. How many times will the following code print "Welcome to Python"?

```
count = 0
while count < 10:
    print("Welcome to Python")
```

- A. 8
- B. 9
- C. 10
- D. 11
- E. infinite number of times

\_\_\_\_\_ 25. How many times is the print statement executed?

```
for i in range(10):
    for j in range(10):
        print(i * j)
```

- A. 100
- B. 20
- C. 10
- D. 45

26. Fill in the code to complete the following function for computing factorial. (5 pts)  
( $N! = N * (N-1)!$ )

```
def factorial(n):
    if n == 0: # Base case
        return 1
    else:
        return _____ # Recursive call
```

27. Answer the following questions based on the 3 lists provided: (10 pts)

a. `len(pi)` \_\_\_\_\_

b. `len(Q)` \_\_\_\_\_

c. `len(Q[1])` \_\_\_\_\_

d. `Q[0]` \_\_\_\_\_

e. `Q[0 :1]` \_\_\_\_\_

f. `Q[0][1]` \_\_\_\_\_

e. `Q[1][0]` \_\_\_\_\_

f. What is `message[9:15]` \_\_\_\_\_

g. What is `message[::5]` \_\_\_\_\_

h. What is `pi[pi[2]]` \_\_\_\_\_

```
pi = [3,1,4,1,5,9]
Q = [ 'pi', "isn't", [4,2] ]
message = 'You need parentheses for chemistry !'
```

28. What will be displayed by the following code? (3 pts)

```
def f1(x = 1, y = 3):
    x = x + y
    y += 1
    print(x, y)
```

`f1()`

29. What will be displayed by the following code? (3 pts)

```
def f1(x = 1, y = 2):
    x = x + y
    y += 1
    print(x, y)
```

`f1(2, 4)`

30. What will be displayed by the following code? (3 pts)

```
def f1(x = 1, y = 2):
    x = x + y
    y += 1
    print(x, y)
```

**f1 (y = 3, x = 4)**

**(6 pts)**

**31.** Write a function called `printGrade` that takes the score and prints the grade base on the score. For example score greater than or equal to 90 is an A, score greater than or equal to 80 is a B and etc...

```
#print grade for the score
def printGrade(score): # complete the function
```

**(20pts)**

**32.** Design a class named `Circle` to represent a circle object. The class contains:

- One data filed named `radius`.
- A constructor that creates a circle with the specified radius. The default value is 1.
- A method named `getPerimeter()` that returns the perimeter of the circle
- A method named `getArea()` that returns the area of the circle
- A method named `setRadius()` that sets the radius of et circle

Draw a UML diagram for the class, and then implement the class. Write a test program that creates two `Circle` objects. – one with radius of 5 and the other with radius 7.5. Display the radius, area, and perimeter of each circle in this order.

## Homework #1 - CSE194

### Getting started with Python on your own machine

*Background: use version 2.7.x*

All you'll need to get started is the Python programming language. **We are using Python version 2.7.3.**

**Warning!** Do NOT use the newer-sounding version Python 3.2.3 or any 3.x version.

#### **What's this homework about?**

The goal for this first program is to introduce you to a few things:

- Getting started with the **python** computer language, which we'll use in CSE194.
- Using python's **IDLE** editor to write and save plain-text files (python programs).
- **Submitting files** to the Blackboard

#### *Installing Python on your own machine!*

If you're on your own machine, the first thing you'll want to do is install Python

#### *Opening up python - on your machine*

- One straightforward way to start up Python and IDLE, Python's built-in editor, is to download a Python (.py) file to your Desktop and then right-click it. Try that approach through these steps:
- **Right-click** and choose **Download file as** or **Save target as** in order to download the file named [hw1.py](#)

**Mac users:** If you have a one-button mouse, you can *control-click* to download on Macs.

- On your desktop or in the download folder, you should see the newly-downloaded file named `hw1.py`. You may not see the `.py` extension, depending on your computer's settings, especially under Windows.
- Right-click on the file's icon and choose `Open with...` and then IDLE. Under Windows, there may be an `Open in IDLE` option from the right-click context menu.
- You should see two windows: one should be an editing window with the contents of the `hw1.py` file -- a python program. You should also see a "Python Shell" window.
  - If you **don't** see the "Python Shell" window, go to the "Run" menu and choose "Python Shell" and it will appear.
  - If you don't see the "Editor" window with the starter code, go to the "File" menu, choose "Open," and then navigate to the `hw1.py` file to open it.

## Trying out python at the python shell

- First, arrange your windows so that you can see the shell, and editor windows
- Then, click on Python's interactive window, the "shell."
- If you ran the python program in `hw1.py`, you will see a prompt asking for your name -- try it! If you haven't run it yet, wait a moment and try out some commands in the shell.
- If nothing is currently running, you'll see the *prompt* of three greater-than signs `>>>`.
- In general, this "shell" is an area for experimenting with the python language. The "prompt" of greater-than signs `>>>` tells you that python is ready to go. You might try `6*7` as a first computation at the prompt.
- Try computing a googol (ten to the hundredth power). The power operator in python is two asterisks `**`. So, you would type at the prompt

```
10**100
```

Admittedly not too enlightening, though the `L` at the end of the result is python's indication that this is a `Long` number.

## Running a program from a file

- Click on the *editor* window to return to the program in the `hw1.py` file.
- You don't need to understand this code - yet. But, there are a few comments that explain a bit of what's happening.
- Take a moment to see if you can guess its behavior: what will the program reply, when different names are typed in?
- To run the program, hit the `F5` key. This is the same as choosing "Run Module" from the "Run" menu.
- The shell window will automatically come up and will start running the program. This program first asks you for your name, and the cursor will be in the right spot to start typing. (The python shell is a bit picky about where the cursor is -- if you move it and start typing, odd things might happen.)
- Type in your name, hit `Enter`, and the program will finish running with a follow-up message.
- You can try again by repeating these steps. Also, you can edit the program to change its behavior.

- To see how python handles errors, you might try removing one of the two equals signs in the `elif` line, so that it reads `elif name = 'Faye'`. Then hit `F5` to run -- it will tell you there is a "syntax error" and give you a chance to fix it.
- **Questions? Problems?** Post them on "Discussion Board" on Blackboard
- For this *program 1*, however, all that's left is to **submit** your `hw1.py` file. It's completely OK if you've changed the file - in fact, I encourage you to improve the dialog in any way you'd like! But that's optional - the goal with this problem is to be sure that you can edit and run a Python program!

### *Submitting your hw1.py file*

- To submit your file, you will want to go to the [myasucourses.asu.edu](http://myasucourses.asu.edu) and then go to CSE194 course.
- You will have to login to Blackboard with your ASURITE name and password. Let me know if you have any questions.

## Homework # 2 – CSE194

Due: Thursday Jan. 31<sup>st</sup>

maximum points: 20

Suppose you want to develop a program to play a lottery. The program randomly generates a two-digit number, prompts the user to enter a two-digit number, and determines whether the user wins according to the following rules:

1. if the user's input matches the lottery in the exact order, the award is \$10,000.
2. if all the digits in the user's input match all the digits in the lottery number, the award is \$3,000
3. if one digit in the user's input matches a digit in the lottery number, the award is \$1,000.

### Hints:

- to create a two digit random number use: `random.randint(0,99)`
- to get the digits use the integer division and the mod function (%)
- in the program you will use selection only (if, elif, and , or, ...)
- **Questions? Problems?** Post them on "Discussion Board" on Blackboard

### *Submitting your hw2.py file*

- Please make sure you have your name and the description of the program as comments at the beginning of the file (-4 points if you do not include this)
- You will have to login to Blackboard with your ASURITE name and password. Let me know if you have any questions.

## **Homework # 3 – CSE194**

Due on: February 21st

maximum points: 20

Suppose you want to develop a program that translates English sentences into Pig Latin.

Pig Latin is a made up language in which each word of a sentence is modified, in general, by moving the initial sound of the word to the end adding an “ay” sound. For example, the word “happy” would be written and pronounced “ appyhay” and the word “birthday” would become irthdaybay. Words that begin with vowels (aeiouy) simply have a “yay” sound added on the end, turning the word “enough” into enoughyay.

Write a program that takes a word from the user and translates it to pig Latin. The program repeat if the user wants to translate another sentence.

### **Hints:**

- Writing a definition (def) will help to break the program into smaller tasks. For example a definition firstVowel to check if the word starts with a vowel.
- **Questions? Problems?** Post them on “Discussion Board” on Blackboard

### ***Submitting your hw3.py file***

- Please make sure you have your name and the description of the program as comments at the beginning of the file (-4 points if you do not include this)
- You will have to login to Blackboard with your ASURITE name and password. Let me know if you have any questions.



## Homework # 4 – CSE194

Due on: March 7th

maximum points: 20

### Converting Decimals to hexadecimal:

In this programming assignment you are writing a program that converts a decimal number to hexadecimal.

Hexadecimal numbers was introduced the first week which often used in computer system programming. To convert a decimal number to a hexadecimal number, you have to divide it by 16 until the quotient is 0. The hexadecimal digits include decimal digits 0,1,2, 3, 4, 5, 6, 7, 8, 9, plus A which is the decimal value 10, B which is the decimal value 11, C which is 12, D which is 13, E which is 14, and F which is 15.

For example, the decimal number 123 is 7B in hexadecimal.

Your program should prompt the user to enter a decimal number and converts it into a hex number as a string.

### Hints:

Start by having a variable `hexString`, initially as an empty string. Use a function called *decimalToHex* to convert a decimal number to a hex number as a string. The function gets the remainder (%) of the division of the decimal integer by 16. The remainder is converted into a character by invoking the *toHexChar* function.

Dividing the decimal number by 16 removes a hex digit from the number. The function repeatedly performs these operations in a loop until the quotient becomes 0.

The *tohexChar* function converts a `hexValue` between 0 and 15 into a hex character. If `hexValue` is between 0 to 9, it is converted to a character. For example: `chr(hexValue + ord('0'))`. For example, if `hexValue` is 5, `chr(hexValue + ord('0'))` returns 5. Similarly, if `hexValue` is between 10 and 15, it is converted to `chr(hexValue - 10 + ord('A'))`. For example, if `hexValue` is 11, `chr(hexValue - 10 + ord('A'))` returns B.

- **Questions? Problems?** Post them on “Discussion Board” on Blackboard

### *Submitting your hw4.py file*

- Please make sure you have your name and the description of the program as comments at the beginning of the file (-4 points if you do not include this)
- You will have to login to Blackboard with your ASURITE name and password. Let me know if you have any questions.

## Homework # 5- CSE194

Due on: March 28<sup>th</sup> by 2:00PM

maximum points: 20

Your programming assignment is to implement a Coin class.

The Coin Class has three methods:

- The `__init__` method : This method initializes the sideup data attributes with 'Heads'
- The toss method: the toss method generates a random number in the range 0 through 1. If the number is 0, teh sideup is set to 'Heads'. Otherwise , sideup is set to 'Tails'
- The `getSideUp` method: return the value referenced by sideup

You can your program with the following main function:

```
# the main function for testing the Coin class
def main():
    # create an object from the Coin class
    myCoin = Coin()

    #Display the side of the coin that is facing up
    print("This side is up: ", myCoin.getSideUp())

    # Toss the coin
    print("I am going to toss the coin 10 times: ")
    for count in range(10):
        myCoin.toss()
        print(myCoin.getSideUP())
```

- **Questions? Problems?** Post them on "Discussion Board" on Blackboard

### *Submitting your hw5.py file*

- Please make sure you have your name and the description of the program as comments at the beginning of the file (-4 points if you do not include this)
- You will have to login to Blackboard with your ASURITE name and password. Let me know if you have any questions.

## Homework # 6- CSE194

**Due on: April 18 by 2:00PM**

maximum points: 20

Your programming assignment is to implement a Loan class:

Consider applying to get a loan. A specific loan can be viewed as an object of a **Loan** class. The interest rate, loan amount, and loan period are its data properties, and computing monthly payment and total payment are its methods.

When you buy a car, a loan object is created by instantiating the class with your loan interest rate, loan amount, and loan period. You can then use the methods to find the monthly payment and total payment of your loan.

Here is the UML class diagram (Unified Modeling language) for the loan class:

<b>Loan</b>
<pre>-annualInterestRate: float -loanAmount: float -borrower: str</pre>
<pre>Loan(annualInterestRate: float, numberOfYears: int, loanAmount float, borrower: str) getAnnualInterestRate(): float getNumberOfYears() : int getLoanAmount() : float getBorrower() : str setAnnualInterestRate(annualInterestRate: flaot) : None setNumberOfYears(numberOfYears : int) : None setLoanAmount( loanAmount: float) : None setBorrower (borrower : str) : None setMonthly Payment() : float getTotalPayment() : float</pre>

- **Questions? Problems?** Post them on “Discussion Board” on Blackboard

### *Submitting your Loan.py file*

- Please make sure you have your name and the description of the program as comments at the beginning of the file (-4 points if you do not include this)
- You will have to login to Blackboard with your ASURITE name and password. Let me know if you have any questions.

## Homework # 7- CSE194

Due on: Tuesday April 30<sup>th</sup>

maximum points: 20

### Lists

Your programming assignment is to write a program that counts the occurrences of each letter among 100 letters, in a list of characters.

1. Write a function called `createList` that generates 100 lowercase letters randomly and assigns them to a list named `myChars`. You can obtain a random letter by using the `getRandomLowerCaseLetter ()` function in the `RandomCharacter` class. For example: `myChars.append(RandomCharacter.getRandomLowerCaseLetters ())`

This function returns the list that has 100 random lower case letters

2. Write a function called `displayList` that takes the list, `myChars`, and displays the characters from the list on the screen with 20 characters per line.

3. Write a function called `countLetters` that takes the list, `myChars`, and counts the occurrences of each letter in the list. To do so, you need to create a list named `counts` that has 26 int values, each of which counts the occurrence if a letter. That is `counts[0]` counts the number of times a appears in the list, `counts[1]` keeps a count of the number of times b appears, and so on.

4. Write a function `displayCounts` that prints the occurrence of each letter.

For example: 5 a 3 b 4 c 5 d

You should print 20 characters per line.

- **Questions? Problems?**

### *Submitting your Hw7.py file*

- Please make sure you have your name and the description of the program as comments at the beginning of the file (-4 points if you do not include this)
- You will have to login to Blackboard with your ASURITE name and password. Let me know if you have any questions.

## Lab 1

### Trying out the Python interpreter (or shell)

There's no file nor anything to hand in for this lab-- just open Python's shell and try out the Python commands below...

The Python shell is the window in which programs execute. You will see a prompt with three greater-than signs at which you can enter short snippets of Python in order to try things out:

```
>>>
```

Arithmetic with numbers, lists, strings, booleans, ...

To get started, try a few arithmetic, string, and list expressions in the Python interpreter, e.g.,

```
>>> 40 + 2
42
```

```
>>> 40 ** 2
1600
```

```
>>> 40 % 2
0
```

```
>>> 'hi there!'
hi there!    (notice Python's politeness!)
```

```
>>> 'who are you?'
who are you?    (though sometimes it's a bit touchy.)
```

```
>>> L = [0,1,2,3]    You can label data (here, a list) with a name (here, the name L)
(no response from Python)
```

```
>>> L
[0, 1, 2, 3]    You can see the data (here, a list) referred to by a name (here, L)
```

```
>>> L[1:]
[1, 2, 3]    You can slice lists (here, using the name L)
```

```
>>> L = 42    You can reassign the name L to another value, even of a different type—now, L names the integer
42, instead of the list it used to represent.
(no response from Python)
```

```
>>> L == 42    Two equals are different than 1! This tests for equality.
True
```

```
>>> L != 42    This tests for "not equal."
False
```

## Errors or Exceptions

Mistakes are unavoidable! So, you'll encounter Python errors. They're sometimes called exceptions, as well.

Perhaps the most important habit we hope you'll pick up when you are programming is: if an error happens, consider it an opportunity, not a problem! An error is a chance to (1) improve your intuition about how computation works, i.e., the "machine's mindset," (2) improve the software you're developing (or your understanding of it), and (3) build on your debugging skills!

So, let's create some!

Give yourself two minutes. In that time, see how many of these Python exceptions you can cause!

If you create others, all the better --:

- NameError (an unrecognized variable!)
- TypeError (try slicing an integer for example!)
- ZeroDivisionError (perhaps clear from its name)
- SyntaxError (the error that kittens most often produce when walking over the keyboard)
- IndexError (try an out-of-bounds index into a sequence)
- OverflowError

Remember that integers won't overflow -- if they get too big to fit in memory, they'll simply crash Python. Thus, to obtain this error you'll need to use floating-point values in some mathematical expression that produces large values!

### Lists! Challenges with slicing and indexing

This problem will exercise your slicing-and-indexing skills.

First, use the File/New Window menu options within the IDLE shell you already have running -- this should create a blank editor window.

To do: Copy the following lines into the editor window:

```
#
# Name:
#
# Lab1.py (Lab 1, part 1)
# slicing and indexing challenges
#

pi = [3,1,4,1,5,9]
e = [2,7,1]
```

```
# Example problem (problem 0):  
# Creating the list [2,5,9] from pi and e  
answer0 = [ e[0] ] + pi[-2:]  
print answer0
```

Then save the contents under the name lab1.py. You need the .py extension to take advantage of IDLE's colorful syntax highlighting.

After the initial comment, this code defines the lists pi and the list e. These lists will be recognized by the Python shell when you hit F5. In addition, the example problem's code will run and print, allowing you to confirm that answer0 is, indeed, the list [2, 5, 9].

The challenge: composing new lists from pi, e, and list operations

The problems below ask you to create several lists using only the list labeled pi, the list labeled e, and these list operations:

- list indexing such as pi[0]
- list slicing such as e[1:]
- skip-slicing such as pi[6:4:-1]
- list concatenation, +, such as pi[:1] + e[1:] (do not use + to add values numerically)
- the list-making operator, [, ] for example: [ e[2], e[0] ]

# CSE194 – Lab 2 – Spring 2013

```
# -----  
# AUTHOR: (Put your name here)  
# FILENAME: Lab2.py  
# SPECIFICATION: This program is for practicing the use of if-else conditional  
# expressions and boolean operators. It also reviews some previous topics.  
# It has two parts:  
# 1. It asks the user to enter two integers, and compares two numbers  
# 2. Find if a year is leap year or not  
  
#Part 1: you need to compare two integers, and output the smaller one  
  
# Prompt the user to enter two integer numbers, store them in two variables: x and y  
  
# Compare the value of x and y, if x is smaller or equal to y, display x; otherwise, display  
  
# Part2: find if a year is a leap year or not  
# a year is leap year if it is divisible by 4 but not by 100 or if it is divisible by 400  
# ask the user to enter a year  
# Display the result
```



## Lab 3 – Spring 2013 – CSE194

```
#-----
# AUTHOR:      (Put your name here)
# FILENAME:    Lab3.py
# SPECIFICATION: This program is for practicing Repetition - the for & while loops.
# You are asked to write a program that randomly generates an integer between 0 and 100, inclusive.
# The program prompts the user to enter numbers continuously until it matches the randomly
# generated number. For each user input the program reports whether it is too low or too high, so the
# user can choose the next input intelligently.
#
# INSTRUCTIONS: Read the following code skeleton and add your own code
#               according to the comments. Ask for help and/or clarification if you need it
#
#-----*/

# import the random module

#Generate a random number between 0 and 100 to be guessed

# prompt the user to guess the number
# write a while loop to see if teh guess is equal to the number
# chek the number against the guess for low or high

import random
number = random.randint(0,100)
count =0
print("Guess a magic number between 0 and 100")

guess = -1
while guess != number and count <5:
    guess = input("Enter your guess: ")
    count +=1
    if guess == number:
        print("yes, the number is" , number)
    elif guess > number:
        print("too high")
    else:
        print ("too low")

if (count == 5):
    print ("The guessing number was: ", number)
```

## Lab 4 – CSE194 – Spring 2013

**# Name: Please put your name here**

**# File name : Lab4.py**

**# date: 2/7/2013**

**# This lab gives you practice with functions**

**# 1. Write a function definition called max that return the maximum of two numbers**

```
def max(num1, num2):
```

```
# complete the code here
```

```
def main():
```

```
    i= 5
```

```
    j = 2
```

```
    k = max(i, j) # call the max function
```

```
    print ("The larger number of" , i, " and", j, "is" , k)
```

```
main() # call the main function
```

**# 2. Write a function definition that finds the sum of integers between two numbers**

```
def sum(i1, i2):
```

```
# complete the code here
```

```
#
```

```
def main():
```

```
    print ("Sum from 1 to 10 is" , sum(1, 10))
```

```
    print ("Sum from 20 to 37 is", sum(20, 37))
```

```
    print("Sum from 35 to 49 is", sum(35, 49))
```

```
main() # call the main function
```

**# 3. Write a function definition that returns a grade based on a score; for example**

**# returns A if the score is greater or equal to 90, B if the score is between 80 to 89 and etc**

```
def getGrade(score):
```

```
# complete the code here
```

```
# define the main by asking the user to enter a score and the print the grade  
# by calling the getGrade function
```

```
# call the main
```

## CSE194 – Lab 5- Spring 2013

```
# Name: Please put your name here
# File name : Lab5.py
# date: 2/7/2013
# This lab gives you practice with loops & selection

# Suppose you are asked to develop a program for a first grader to practice
# subtraction. The program randomly generates two single digit integers,
# number1 and number2, with number 1 >= number2 and asks the student a question
# such as "What is 9 - 2? " after the student enters the answer, the program
# displays a message indicating whether it is correct.
# The program lets the student to enter a new answer until it is correct.

# the program may work as follows:
# Step 1: Generate two single digit integers for number1 and number2
# step 2: If number 1 < number2 , swap number1 with number2
# step 3: Prompt the student to answer, "What is number1 - number2?"
# step 4: check the student's answer and while the answer is not correct keep asking for the correct
answer

import random
#Step 1: Generate two single digit integers for number1 and number2

#Step 2: If number 1 < number2 , swap number1 with number2

#Step 3: Prompt the student to answer, "What is number1 - number2?"

# step 4: check the student's answer and while the answer is not correct keep asking
# for the correct answer - while number1- number2 != answer repeat the question
```

## Lab 6- CSE194 – Spring 2013

```
# Name: Please put your name here
# File name : Lab6.py
# date: 2/19/2013
# This lab gives you practice with loops & selection
```

```
# When you invoke a function with arguments, each argument's reference is passed
# by value to the parameter in the function
```

**# Example:** Try to run this function

```
def main():
    x =1
    print("Before the call, x is", x)
    increment(x)
    print("After teh call: x is", x)

def increment(n):
    n += 1
    print("\tn inside the function is" , n)
main() # call the main function
```

**# Question 1 :** explain the output

```
x =4
y = x
print id(x) # the reference of x
print id(y) # the reference of y is the same as the reference x
```

**# Question 2:** Explain the output

```
y = y +1 # y now points to a new int object with value 5
print id(y)
```

**# Question 3:** Explain the output

```
def getMax(value1, value2, max):
    if (value1 > value2):
        max = value1
    else:
        max = value2

def main():
    max = 0
    getMax(1, 2, max)
    print(max)
main()
```

**# Question 4:** Show the contents of the max just before the function max is invoked,  
# just as max is entered, just before max is returned, and right after max is returned.

```
def main():
    times = 4 # initialize times
    #Invoke the function nPrint
    nPrint("Welcome to CSE194!" , times)
    print"Afetr the call, variable times is", times
#Print the message n times
def nPrint(message, n):
    while n>0:
        print"n= ", n
        print(message)
        n -= 1
main()
```

## Lab 7 – CSE 194 – Spring 2013

```
#  
# Name: Please put your name here  
# File name : Lab7.py  
# date: 2/21/2013  
# This lab gives you practice with functions, scope of variables and parameters
```

```
# The scope of variables  
# The scope of a variable is the part of the program where the variable can be referenced
```

**# Example 1:** Explain the output

```
globalVar = 1  
def f1():  
    localVar = 2  
    print(globalVar)  
    print(localVar)  
f1()  
print(globalVar)  
print(localVar)
```

**# Example 2:** Explain the output

```
x = 1  
def f1():  
    x = 2  
    print(x)  
f1()  
print(x)
```

**# Example 3:** Explain the output if x = 5 and then when x = 0

```
x = input("Enter a number: ")  
if x > 0:  
    y = 10  
print(y)
```

**# Example 4:** Explain the output

```
sum = 0  
for i in range(5):  
    sum += i  
print(i)
```

**#Example 5:** You can create a variable in a function and use it outside  
# the function by using global keyword, as shown in the following example

```
x =1
def increase():
    global x
    x = x +1
    print(x)
increase()
print(x)
```

### # Example 6

# Default arguments

# Python allows you to define functions with default arguments values. The default values  
# are passed to the parameters when a function is invoked without the arguments

```
def printArea(width =1, height =2):
    area = width * height
    print "width:" , width, '\t', "height:", height, '\t', "area:" ,
area
printArea() # Default arguments width =1 and height =2
printArea(4, 2.5) # Positional arguments width = 4 and height =2.5
printArea(height = 5, width =3) #keyword arguments width and height
printArea(width=1.2) # default height =2
printArea(height = 6.2) # default width =1
```

# Write a function definition called GCD that prompts

# the user to enter two integers and displays their greatest common divisor.

```
# complete the code here
def gcd(num1, num2):
```

```
def main():
    n1 = input("Enter the first integer: ")
    n2 = input ("Enter the second integer: ")
    print ("The greatest common divisor fro", n1, "and", n2, "is",
gcd(n1,n2))
```

```
main() # call the main function
```



## Lab 8 – CSE194

Maximum points: 10

Write a program that prints a menu with 7 options:

Option 1: Asks the user to input a new list

Option 2: prints the current list

Option 3: Finds the average of the values in the list

Option 4: finds the standard deviation for the list

Option 5: find the minimum value in the list

Option 6: finds the maximum in the list

Option 7: quits

- **Questions? Problems?** Post them on “Discussion Board” on Blackboard

### *Submitting your lab8.py file*

- Please make sure you have your name and the description of the program as comments at the beginning of the file (-4 points if you do not include this)
- You will have to login to Blackboard with your ASURITE name and password. Let me know if you have any questions.

## **Lab 9 – CSE194**

Maximum points: 10

The Circle class: Design a class named Circle to represent a circle object. The class contains:

- One data field named radius.
- A constructor that creates a circle with the specified radius. The default value is 1.
- A method named getPerimeter() that returns the perimeter of the circle
- A method named getArea() that returns the area of the circle
- A method named setRadius() that sets the radius of the circle
- Draw a UML diagram for the class, and then implement the class. Write a test program that creates two Circle objects. – one with radius of 5 and the other with radius 7.5. Display the radius, area, and perimeter of each circle in this order.

### ***Submitting your Circle.py file***

- Please make sure you have your name and the description of the program as comments at the beginning of the file (-4 points if you do not include this)
- You will have to login to Blackboard with your ASURITE name and password. Let me know if you have any questions.

## **Lab 10 - CSE194**

Maximum points: 10

The Rectangle class: Design a class named Rectangle to represent a rectangle object. The class contains:

- Two data fields named width and height.
- A constructor that creates a rectangle with the specified width and height. The default values are 1 and 2 for the width and height, respectively.
- A method named getPerimeter() that returns the perimeter of the rectangle
- A method named getArea() that returns the area of the rectangle
- Draw a UML diagram for the class, and then implement the class. Write a test program that creates two Rectangle objects. – one with width 4 and height 5 and the other with width 3.5 and height 34.5. Display the width, height, area, and perimeter of each rectangle in this order.

### ***Submitting your Rectangle.py file***

- Please make sure you have your name and the description of the program as comments at the beginning of the file (-4 points if you do not include this)
- You will have to login to Blackboard with your ASURITE name and password.

## Lab 11 - CSE194

Maximum points: 10

**The Rational class:** Design a class named Rational class for representing and processing rational numbers.

A rational number has a numerator and a denominator in the form  $a/b$ , where  $a$  is the numerator and  $b$  is the denominator. For example,  $1/3$ ,  $3/4$  and  $10/4$  are rational numbers.

A rational number cannot have a denominator of 0, but a numerator of 0 is fine. Every integer  $i$  is equivalent to a rational number  $i/1$ . Rational numbers are used in exact computations involving fractions- for example,  $1/3 = 0.33333\dots$ . This number cannot be precisely represented in floating-point format using data type float. To obtain the exact result, we must use rational numbers.

A rational number can be represented using two data fields: numerator and denominator. You can create a rational number with a specified numerator and denominator. You can add, subtract, multiply, divide, and compare rational numbers.

Design the class and use the following program Lab11.py to test your Rational class.

```
# Name: Please put your name here
# File name : Lab11.py
# date: 3/26/2013
# This lab gives you practice with Classes & Objects

from Rational import Rational
# create and initialize two rational numbers r1, and r2
r1 = Rational(6, 8)
r2 = Rational(2,3)
print r1.getDenominator()
print r1.getNumerator()
r1.__str__()
r2.display()

# add two rational numbers and display the result
r3= r1.__add__(r2)
print r1.display() + "+" + r2.display() + "=", r3.display()

#print(r1, "-", r2 "=", r1.__sub__(r2))
#print(r1, "*", r2 "=", r1*r2)
#print(r1, "/", r2 "=", r1/r2)
```

### *Submitting your Rational.py file*

- Please make sure you have your name and the description of the program as comments at the beginning of the file (-4 points if you do not include this)
- You will have to login to Blackboard with your ASURITE name and password.

## Lab 12 – CSE194 – Practice Strings

Maximum points: 10

- A string is an Object, and it is immutable. Its contents cannot be changed.
- You can use the Python functions len, min, and max to return the length of a string, and the minimum and maximum elements in a string
- You can use the index operator [] to reference an individual character in a string.
- You can use the concatenation operator + to concatenate two strings, the repetition operator \* to duplicate strings, the slicing operator [:] to get a substring and the in and not in operators to check whether a character is in a string.
- The comparison operators (==, !=, <, <=, >, and >=) can be used to compare two strings.
- You can use a for loop to iterate all characters in a string
- You can use the methods such as endswith, startswith, isalpha, islower, isupper, lower, upper, find, count, replace, and strip on a string object.

Suppose that s1, s2, s3, and s4 are four strings, given as follows:

```
s1 = "Welcome to Python"  
s2 = s1  
s3 = "Welcome to Python"  
s4 = "to"  
s1 == s2
```

What are the results of the following expressions?

- |                           |                             |
|---------------------------|-----------------------------|
| a. s1 == s2 _____         | l. 4 * s4 _____             |
| b. s2.count('o') _____    | m. len(s1) _____            |
| c. id(s1) == id(s2) _____ | n. max(s1) _____            |
| d. id(s1) == id(s3) _____ | o. s1[-4] _____             |
| e. s1 <= s4 _____         | p. min(s1) _____            |
| f. s2 >= s4 _____         | q. s1.lower() _____         |
| g. s1 != s4 _____         | r. s1.startswith("o") _____ |
| h. s1.upper() _____       | s. s1.endswith("n") _____   |
| i. s1.find(s4) _____      | t. s1 + s1 _____            |
| j. s1[4] _____            | k. s1[4:9] _____            |

**Lab 12:** Write a program that checks whether a string is a palindrome.

A string is palindrome if it reads the same forward and backward. The word “mom”, “madam” and “noon” for instance, are all palindromes.

Write a function called `isPalindrome` that takes a string and return true if the string is a palindrome.

Here is the main function:

```
def main():
    #prompt the user to enter a string
    s = input("Enter a string: ")
    if isPalindrome(s):
        print(s + " is a palindrome")
    else:
        print(s + " is not a palindrome")
```

### ***Submitting your Lab12.py file***

- Please make sure you have your name and the description of the program as comments at the beginning of the file (-4 points if you do not include this)
- You will have to login to Blackboard with your ASURITE name and password.

## Lab 13 – CSE194 – Practice Functions

Maximum points: 10

**Lab 12:** Write a program that calculates the following:

- The area of a circle (you need to use your Circle class )
- The circumference of a circle
- The area of a rectangle (you need to use your Rectangle class)
- The perimeter of a rectangle

You already have the Circle class and Rectangle class from the previous labs.

The program should provide a menu to the user with the following options:

- 1) Area of a circle
- 2) Circumference of a circle
- 3) Area of a rectangle
- 4) Perimeter of a rectangle
- 5) Quit

Ask the user to enter the choice and based on user input call the appropriate function in the class Circle or the class Rectangle. The program keeps displaying the menu until user enters option 5.

### *Submitting your Lab13.py file*

- Please make sure you have your name and the description of the program as comments at the beginning of the file (-4 points if you do not include this)
- You will have to login to Blackboard with your ASURITE name and password.

## Lab 14- CSE194

Maximum points: 10

### Checking for Palindromes Recursively:

A string is a palindrome if it reads the same forward and backward. The words “mom”, “madam” and “noon” for instance, are all palindromes.

Write a program that prompts the user to enter a string and reports whether the string is palindrome or not.

Write 2 functions, isPalindrome and recursivePalindrome, one using loops (iteration) and the second function using recursion

**Hint:** it is common design technique in recursive programming to define a second function that receives additional parameters. Such a function is known as a recursive helper function.

Helper functions are very useful in designing recursive solutions for problems involving strings and lists.

- **Questions? Problems?**

### *Submitting your lab14.py file*

- Please make sure you have your name and the description of the program as comments at the beginning of the file (-4 points if you do not include this)
- You will have to login to Blackboard with your ASURITE name and password. Let me know if you have any questions.



## Quiz 1 - If statements

Last Name (Print) \_\_\_\_\_ First Name \_\_\_\_\_

1. What value is contained in the variable length after the following statements are executed?

```
length = 5
length += 3
length = length * 2
```

answer: \_\_\_\_\_

2. What is the result of 2/4 when evaluated as a Python expression \_\_\_\_\_

Explain why?

3. Write the output generated by the following piece of code:

```
grade = 45
if( grade >= 70 ):
    print('passing')
if( grade < 70 ):
    print('dubious')
if( grade < 60 ):
    print('failing')
```

4. Write the output generated by each code segment given these initializations of j and x:

```
j = 8
x = -1.5
```

a.

```
if( x < -1.0 ):
    print( "true" )
else:
    print( "false" )
```

b.

```
if( x >= j ):
    print( "x is high" )
else:
    print( "x is low" )
```

c.

```
if( x <= 0.0 ):
    if( x < 0.0 ):
        print( "neg" )
else:
    print( "zero" )
else:
    print( "pos" )
```

5. Write a Boolean expression that prints true if number is odd and false if number is even.

**Answers:**

1. 16

2. 0 it is an integer division

3.

dubious  
failing

4.

a. true  
b. x is low  
c. neg

5.

```
if( number % 2 == 0):
```

```
    print false
```

```
else:
```

```
    print true
```

## Quiz 2 - Lists & Strings - CSE194

Last Name (Print) \_\_\_\_\_ First Name \_\_\_\_\_

1. Answer the following questions based on the 3 lists provided:

a. len(pi) \_\_\_\_\_

b. len(Q) \_\_\_\_\_

c. len(Q[1]) \_\_\_\_\_

d. Q[0] \_\_\_\_\_

e. Q[0:1] \_\_\_\_\_

f. Q[0][1] \_\_\_\_\_

e. Q[1][0] \_\_\_\_\_

f. What is message[9:15] \_\_\_\_\_

g. What is message[:5]

h. What is pi[pi[2]] \_\_\_\_\_

i. pi[0] \* (pi[2] + pi[4]) \_\_\_\_\_

```
pi = [3,1,4,1,5,9]
Q = [ 'pi', "isn't", [4,2] ]
message = 'You need parentheses for chemistry !'
```

2. s = 'Welcome to Python'

a. what is: s[0:7] \_\_\_\_\_

b. what is: s[8:12] \_\_\_\_\_

c. what is: s[11:16] \_\_\_\_\_

3. Show the result of the following code:

```
sum = 2+3
print(sum)
s = '2' + '3'
print(s)
```

**Answers:**

1.

- a. 6
- b. 3
- c. 5
- d. pi
- e. ['pi']
- f. i
- e. i
- f. parent
- g. yeah cs!
- h. 5
- i. 27

2.

- a. Welcome
- b. to P
- c. Pytho

3.

- 5
- 23

Quiz 3 - Loops - CSE194

Last Name (Print) \_\_\_\_\_ First Name \_\_\_\_\_ Date \_\_\_\_\_

What do these loops print? What do they do?

1.

```
s = 'gattacaaggtaaaatgca'
N = 0
for i in range(0,len(s)):
    if s[i] == 'a':
        N = N + 1
print 'N is', N
```

2.

```
s = 'gattacaaggtaaaatgca'
N = 0
for i in range(0,len(s)):
    if s[i] == 'a' or s[i] == 't':
        N = N + 1
print 'N is', N
```

3.

```
s = 'gattacaaggtaaaatgca'
N = 0
for i in range(1,len(s)):
    if s[i] == 'a' and s[i-1] == 't':
        N = N + 1
print 'N is', N
```

4.

```
s = 'gattacaaggtaaaatgca'
MAX = 0
cur = 0
for i in range(0,len(s)):
    if s[i] == 'a':
        if s[i-1] != 'a':
            cur = 1
        else:
            cur = cur + 1
            if cur > MAX:
                MAX = cur

print 'Max is', MAX
```

**Answers:**

1. N is 9
2. N is 13
3. N is 2
4. Max is 4

Quiz 4 - Strings - CSE194

Last Name (Print) \_\_\_\_\_ First Name \_\_\_\_\_ Date \_\_\_\_\_

Suppose word = "image" and phrase = "protein synthesis" .

A) Determine the values of these Python expression:

1. word[0] \_\_\_\_\_
2. word[2:] \_\_\_\_\_
3. word[:2] \_\_\_\_\_
4. word[::2] \_\_\_\_\_
5. phrase[1:5] \_\_\_\_\_
6. phrase[:-4] \_\_\_\_\_
7. phrase [:8] + word

B) Create Python expressions using word and phrase as above that have these values:

8. 'g' \_\_\_\_\_
9. 'mag' \_\_\_\_\_
10. 'pro' \_\_\_\_\_
11. 'image synthesis' \_\_\_\_\_

C) Suppose that s1, s2, s3 and s4 are four strings, given as follows:

```
s1 = "Welcome to Python"  
s2=s1  
s3="Welcome to Python"  
s4="to"
```

- |                        |                             |
|------------------------|-----------------------------|
| 1. s1 == s2 _____      | 6. s1.endswith("o") _____   |
| 2. s2.count('o') _____ | 7. s1.startswith("w") _____ |
| 3. s1.lower() _____    | 8. S1 + s1 _____            |
| 4. s1[4:8] _____       | 9. s1 != s4 _____           |
| 5. s1.find(s4) _____   | 10. id(s1) == id(s2) _____  |
| 6. len(s1) _____       | 11. id(s1) == id(s3) _____  |

Answers:

A)

1. i
2. age
3. im
4. iae
5. rote
6. protein synth
7. protein image

B)

```
print word[3]
print word[1:4]
print phrase[0:3]
print word + phrase[7::]
```

C)

1. True
2. 3
3. welcome to python
4. ome
5. 8
6. 17
7. False
8. False
9. Welcome to PythonWelcome to Python
10. True
11. True
12. False



## Quiz 5 - CSE194

Last Name (Print) \_\_\_\_\_ First Name \_\_\_\_\_ Date \_\_\_\_\_

**Show the output of the following code:**

```
1.
i=1
for i in range(1, 10):
    if i <=5:
        print i, 'Smaller or equal than 5.\n'
    else:
        print i, 'larger than 5. \n'
```

2.

```
s = "abcdefg"
print s[0:4]
print s[4:]
print s[4:0]
print s[0:2]
print s[0:7:2]
```

3.

```
myList = ["Linux", "Mac OS", "Windows"]
print(myList[0])
print(myList[-1])
print myList[0:2]
myList.append("Android")
for element in myList:
    print element
```

4.

```
def main():
    i=1
    while i<=6:
        print(myfunction(i,2))
        i += 1

def myfunction(i, num):
    line = ""
    for j in range (1, i):
        line +=str(num) + ""
        num *= 2
    return line
main()
```

5.

```
def main():
    i=0
    while i <= 4:
        mystery(i)
        i += 1

def mystery(i):
    line = ""
    while i >=1:
        line += str(i) + " "
        i -= 1

    print (line)
main()
```

6. Write a function called sum that takes two integers and returns the sum of the values between the two integers; For example sum(1, 10) returns 55 and sum(20, 37) returns 513.

Answers:

1.

1 Smaller or equal than 5.

2 Smaller or equal than 5.

3 Smaller or equal than 5.

4 Smaller or equal than 5.

5 Smaller or equal than 5.

6 larger than 5.

7 larger than 5.

8 larger than 5.

9 larger than 5.

2.

abcd

efg

ab

aceg

3.

Linux

Windows

['Linux', 'Mac OS']

Linux

Mac OS

Windows

Android

4.

2

24

248

24816

2481632

5.

1

2 1

3 2 1

4 3 2 1

## Quiz 6 - CSE194

Last Name (Print) \_\_\_\_\_ First Name \_\_\_\_\_ Date \_\_\_\_\_

1. Describe the relationship between an object and its defining class.

An object is an instance of a class.

2. How do you define a class?

```
class ClassName:
```

3. How do you create an object?

```
object = ClassName(arguments)
```

4. What is the name of the initializer method?

```
__init__
```

5. What is the difference between an initializer and a method?

You use the initialize (constructor) to create an object. You use the dot operator (.) to access the methods.

6. What is the syntax for constructing an object? What does Python do when constructing an object?

```
Object = ClassName (arguments)
```

Python automatically assigns each object a unique id for identifying the object at runtime.

```
def sum(num1, num2):  
    result =0  
    for i in range(num1, num2+1):  
        result += i  
    return result  
  
def main():  
    print "Sum from 1 to 10 is", sum(1,10)# call the sum function  
    print "Sum from 20 to 37 is", sum(20,37)  
main() # call the main function
```

## Quiz 7 – CSE 194- Spring 2013

Last Name \_\_\_\_\_ First: \_\_\_\_\_ Date: \_\_\_\_\_

1. What will be displayed by the following code?

```
myList = [1, 5, 5, 5, 5, 1]
max = myList[0]
indexOfMax = 0
for i in range(1, len(myList)):
    if myList[i] > max:
        max = myList[i]
        indexOfMax = i

print(indexOfMax)
```

2. What will be displayed by the following code?

```
myList = [1, 2, 3, 4, 5, 6]
for i in range(1, 6):
    myList[i - 1] = myList[i]

for i in range(0, 6):
    print(myList[i], end = " ")
```

3. What will be displayed by the following code?

```
list1 = [1, 3]
list2 = list1
list1[0] = 4
print(list2)
```

4. What will be displayed by the following code?

```
def f(values):
    values[0] = 44

v = [1, 2, 3]
f(v)
print(v)
```

5. What will be displayed by the following code?

```
def f(value, values):
    v = 1
    values[0] = 44

t = 3
v = [1, 2, 3]
f(t, v)
print(t, v[0])
```

6. What will be displayed by the following code?

```
def f(i, values = []):  
    values.append(i)  
    return values
```

```
f(1)  
f(2)  
v = f(3)  
print(v)
```

7. \_\_\_\_\_ creates a list.

- A. list1 = list()
- B. list1 = []
- C. list1 = list([12, 4, 4])
- D. list1 = [12, 4, 4]
- E. list1 = [1, "3", "red"]

8. Suppose list1 is [3, 4, 5, 20, 5, 25, 1, 3], what is len(list1)?

\_\_\_\_\_

9. Suppose list1 is [1, 3, 2], what is sum(list1)? \_\_\_\_\_

10. Suppose list1 is [1, 3, 2], what is print list1[1]?

11. Suppose list1 is [1, 3, 2, 4, 5, 2, 1, 0], What is list1[:-1]?

Answers:

1. 1

2. [2, 3, 4, 5, 6, 6]

3. [4, 3]

4. [44, 2, 3]

5. 3 44

6. [1, 2, 3]

7. All of the choices (A, B, C, D, E)

8. 8

9. 6

10. 3

11. [1, 3, 2, 4, 5, 2, 1]