2019

CST1101–Problem Solving with Computer Programming, Syllabus, Spring 2019

Douglas L. Moody
CUNY New York City College of Technology

NEW YORK CITY COLLEGE OF TECHNOLOGY/CUNY
DEPARTMENT OF COMPUTER SYSTEMS TECHNOLOGY

CST1101–PROBLEM SOLVING WITH COMPUTER PROGRAMMING
(4 hours – 3 credits)

Instructor:
Name: Prof.
Office:
e-mail:
Office Hours:

Course Description:
This course introduces students to concepts of problem-solving using constructs of logic inherent in computer programming languages. Students study the nature of problems, common solution approaches and analysis techniques. Students use a pseud-coding tool to execute instructions involved in problem solving exercises. These solutions are augmented by a flowchart interpreter to diagram these problem solutions. The second half of the course transitions from problem solving to learning the basics of the computer programming language Python. Both Python scripts and flowcharts enable students to construct solutions to common algorithmic problems. The major emphasis is on teaching the student to identify solutions to a problem and translate them into various forms that will enable the computer to perform some of the steps in a solution of an actual problem instance. These forms include pseudo-coding environment, flowcharting tool, and developing Python scripts that demonstrates the students' knowledge of all the basic programming concepts discussed in problem solving lectures (e.g., data abstraction, conditions, repetitions and code block as methods, functions).

Course Objectives:
Upon successful completion of the course, students should be able to:

1. Demonstrate understanding of the steps required to solve a problem using a computer.
2. Understand different approaches to a problem such as Top-Down, Bottom-Up and Divide and Conquer
3. Demonstrate the ability to recognize patterns and repetition within give problems and algorithms.
4. Demonstrate understanding of flowcharting techniques to solve an algorithm.
5. Understand how basic instruction processing concepts as loops, methods, conditional statements can be used to solve problems.
6. Be familiar with problem solving approaches that emphasize up front analysis, testing and validation
7. Demonstrate understanding of the major programming elements including variables, syntax requirements and language keywords such as While, Break, etc.
8. Install and run the IDLE Python programming environment.
9. Design and implement basic Python scripts.

10. Demonstrate broad problem-solving experience by referring to solutions from a problem bank covered during class.

General Education Outcomes:

- **SKILLS/Inquiry/Analysis:** Students will employ scientific reasoning and logical thinking.

- **SKILLS/Communication:** Students will communicate in diverse settings using oral (both speaking and listening) and visual means.

- **VALUES, ETHICS, RELATIONSHIPS / Professional/Personal Development:** Students will have access to on-line materials and solutions to programming problems and will be required to process those materials and solutions, understand them, use the ideas from them without passing others' ideas as their own.

Prerequisite – CUNY certification in mathematics, reading and writing. General knowledge of a personal computer is needed. Students may enroll in a workshop at the Academic Learning Center, located in the Atrium.

This is an OER (Open Educational Resources) course. All the required reading materials are free. The OER page for the course can be viewed here:
https://openlab.citytech.cuny.edu/cst1101-problemsolvingpython

Software Download (free, online)

- Pathway2Code is the pseudo-coding learning platform to support executing high level natural language commands to solve problems. The platform can be found at www.pathway2code.com. The instructor will provide access information such as class id , username and password.

- Python official site that includes documentation, downloads (IDLE for Python 3.6), news:

  https://www.python.org

- Flowchart interpreter

  http://www.flowgorithm.org/

Required Reading (free, online)

- Think Python, 2nd Edition by Allen B. Downey

  http://greenteapress.com/wp/think-python-2e/

- How to Think Like a Computer Scientist: Interactive Edition

  http://interactivepython.org/runestone/static/thinkcspy/index.html

Recommended reading (free, online)

- Algorithmic Problem Solving with Python by John B. Schneider, Shira Lynn Broschat, and Jess Dahmen.

  

  http://www.eecs.wsu.edu/~schneidj/swan/index.php

- How to Think Like a Computer Scientist by Peter Wentworth, Jeffrey Elkner, Allen B. Downey, and Chris Meyers

  http://www.openbookproject.net/thinkcs/python/english3e/

- Python Bibliotheca: http://www.openbookproject.net/pybiblio/

Tentative Evaluation and Grading Distribution (the exact grade distribution is defined by the instructors teaching individual sections)

Test1	20%
Test2	20%
Assignments	20%
Classwork	10%
Final Exam (cumulative)	30%
	=====
Total	100%

Grade System:

Letter Grade	A	A-	B+	B	B-	C+	C	D	F
Numerical Grade	93-100	90-92.9	87-89.9	83-86.9	80-82.9	77-79.9	70-76.9	60-69.9	<=59.9

The grade distribution follows the information in the NYCCT Student Handbook (p.43).

During the course of the class you are required to follow the NYCCT Academic Integrity Standards described in the Student Handbook (pp.95 – 99)

NYCCT Student Handbook can be downloaded here: http://www.citytech.cuny.edu/current-student/docs/StudentHandbook.pdf.

Tentative schedule

- **Schedule is based upon 3 sections with varying lengths depending upon class and instructor:**

  Phase I - Problem Solving and Pseudo Coding (4-8 weeks)

  Phase II Flowcharting (2-4 weeks)

  Phase III Python Programming (6-10 weeks)

- **the changes in the schedule can be made to address the pace of an individual class**
- **Reading assignment for the topics can be checked on the CST 1101 OER site:** https://openlab.citytech.cuny.edu/cst1101-problemsolvingpython/assignments/

This schedule is to be tuned to the instructors overall time allocated to each phase. In the table, for example, some instructors may cover the material in 2 sessions versus 3.
Tests can be injected where appropriate. Only 24 classes are shown here to allow for tests, reviews, catch-up periods.

Class #	Topic name
1	Course Introduction Class logistics; Introduction: • What is problem solving? Computer problem solving: • Solution = program / algorithm • Well-defined set of steps • Examples of problems solved using sets of steps: cooking recipes, puzzles Computer problem solving Pseudo code examples
2 3	Solutions as Instruction Sets • Type of Directions (Recipe Assembly) • Structuring natural language into specific commands • Instructions and commands • Sequencing of instructions to solve a problem • Use of parameters to enhance command capabilities Classwork: Pathway2Code to create initial solutions of constructing block letters, patterns, musical compositions

4 5	<u>Graphics and Recognizing repetition</u> Computer Graphics • RGB Colors • Coordinate System Using methods to reduce repetitive commands Using loops to execute the same set of commands multiple times Classwork: Pathway2Code to create graphical objects such as flags, soccer fields, snowmen
6 7	<u>Computer Storage</u> Working with the Loop Index inside a loop Creating variables and assignment of values Using Math operators Inputting information Classwork: Pathway2Code to create expand previous projects, accept user input and perform calculations, solve problems based on functions of a loop index.
8 9	<u>Problem Solving Strategies</u> Basic Approaches : Top-Down, Bottom-Up, Divide and Conquer IRPO – Input, Reference, Process , Output description of problem solutions PACT – Problem Definition, Analysis, Coding, Testing – approach to word problems Classwork – Walkthrough problems and apply above strategies
10	<u>Decision Making</u> • Relational operators • Boolean conditional • If Then Else construct Classwork: Pathway2Code to develop solutions where conditional statements are required.
11	<u>Random Numbers (Optional Topic)</u> • How numbers are provided • Random numbers related commands Classwork: Pathway2Code to make graphical outputs based upon randomness (e.g. colors, geometric object location)
12 13	<u>Flowcharting</u> • Purpose – Diagram view • Debugging concepts – Step, Watch, Breakpoint Classwork – Use selected tool to create solutions to IRPO problems and demonstrate tracing / debugging capabilities

14	Python Programming Environment • Module versus Shell - IDLE • Key language syntax issues • Executing Pathyway2code instructions within Python Classwork – Modify existing Pathway2code projects within IDLE environment.
15 16	Data Types – Conversion issues • Implicit data typing • String versus integer for input • Conversion functions - int Classwork – Demonstration of solutions that require inputs of different data types
17	Functions • Returning values • Placement • Global variables
18 19	Loops • While loop • For loop start and step parameters • Break Continue
20 21	Logical Operators • Truth Tables • Embedding logical operators in conditionals
22	Advanced Strings • Substring • Key methods – Uppercase,find
23 24	Lists

Assessment Criteria

For the successful completion of this course a student should be able to:	Evaluation methods and criteria

1. Demonstrate understanding of the steps required to solve a problem using a computer.	Students will describe problem, identify inputs, processes and desired outcomes in laboratory assignments, class work and tests.
2. Demonstrate understanding of flowcharting techniques to solve an algorithm.	Students will solve problems using the flowchart interpreter software and Python 2.7 in laboratory assignments, class work and tests.
3. Understand the best problem-solving approach for a given problem.	Students identify the approach of Top-Down, Bottom-Up, Divide-Conquer given the problem characteristics.
4. Produce graphical outputs with inherent repetition and patterns.	Students will use a pseudo-coding platform to develop graphical patterns containing repetitive graphical objects.
5. Demonstrate the knowledge or Boolean algebra (AND, OR, NOT operations)	Students will solve Boolean algebra problems in laboratory assignments, class work and tests and incorporate these solutions in flowcharts and Python scripts.
6. Demonstrate understanding of the major programming notions: variables, decision statements, repetition/loop statements (both count- and event-controlled), arrays/lists, modules/functions, classes and objects and their use for basic problem solving.	Students will create algorithms for problem solving using the basic programming notions in laboratory assignments, class work and tests.
7. Demonstrate understanding of the two major programming paradigms: procedural and object-oriented.	Students will create new classes and objects of these classes in laboratory assignments, class work and tests.
8. Install and run the IDLE Python programming environment.	To complete homework assignments and practice programming skills outside the college students will install the IDLE Python environment on their own computers.
9. Design and implement basic Python scripts.	Students will use the knowledge of Boolean Algebra, problem solving paradigms and basic programming notions to write Python scripts in laboratory assignments, class work and tests.
10. Demonstrate broad problem-solving experience by referring to solutions from a problem bank covered during class	Students will demonstrate problem-solving ability in laboratory assignments, class work and tests.

General Education Outcomes and Assessment:

Learning Outcomes	Assessment Method
SKILLS/Inquiry/Analysis Students will employ scientific reasoning and logical thinking.	Students will describe problem, identify inputs, processes and desired outcomes in laboratory assignments, class work and tests. Students will solve problems using the flowchart interpreter software and Python in laboratory assignments, class work and tests. Students will identify coding paradigms in Laboratory Assignments, Class work and tests
SKILLS/Communication Students will communicate in diverse settings using oral (both speaking and listening) and visual means.	Students will discuss various problems and approaches towards solving these problems in class
VALUES, ETHICS, RELATIONSHIPS / Professional/Personal Development Students will have access to on-line materials and solutions to programming problems and will be required to process those materials and solutions, understand them, use the ideas from them without passing others' ideas as their own.	Students will learn to respectfully use the code generated by other programmers giving.