UMBC UGC New Course Request: CMSC210 – Advanced Computing

Date Submitted: 3/42020 Proposed Effective Date: 8/23/2020

	Name	Email	Phone	Dept
Dept Chair or UPD	Jeremy Dixon	jdixon@umbc.edu	5-8866	CSEE
Other Contact	Anupam Joshi	joshi@umbc.edu	52733	CSEE
Other Contact	Mohamed Younis	younis@umbc.edu	5-3969	CSEE

COURSE INFORMATION:

Course Number(s)	CMSC210
Formal Title	Advanced Computing
Transcript Title (≤30c)	Advanced Computing
Recommended Course Preparation	
Prerequisite **NOTE:** Unless otherwise indicated, a prerequisite is assumed to be passed with a "D" or better.	CMSC 201 with a C or better
# of Credits Must adhere to the UMBC Credit Hour Policy	3
Repeatable for additional credit?	☐ Yes ☒ No
Max. Total Credits	3 This should be equal to the number of credits for courses that cannot be repeated for credit. For courses that may be repeated for credit, enter the maximum total number of credits a student can receive from this course. E.g., enter 6 credits for a 3 credit course that may be taken a second time for credit, but not for a third time. Please note that this does NOT refer to how many times a class may be retaken for a higher grade.
Grading Method(s)	☒ Reg (A-F) ☒ Audit ☒ Pass-Fail

PROPOSED CATALOG DESCRIPTION (Approximately 75 words in length. Please use full sentences.):

This course strengthens and extends the student's programming and problem-solving skills through the use of advanced programming language constructs, pre-defined libraries, and proper software engineering techniques. Topics include program design, debugging, and testing, source code versioning control, use of a software development environment, data formats, web programming, web data extraction, and data visualization. This is the second course for non-computer science, non-computer engineering majors interested in pursuing further study in applied computing.

RATIONALE FOR NEW COURSE:

a) Why is there a need for this course at this time?
This course is the second course in our applied computing sequence intended for non-computer science, non-computer engineering majors. It has been piloted as a special topics course and is ready to be offered on its own.
b) How often is the course likely to be taught?
We will most likely offer it every spring semester, increasing to every semester as the population increases.
c) How does this course fit into your department's curriculum?
It is the second course in our applied computing sequence intended for non-computer science, non-computer engineering majors.
d) What primary student population will the course serve?

This course will serve non-computer science, non-computer engineering majors, typically at the freshman and sophomore levels.

e) Why is the course offered at the level (ie. 100, 200, 300, or 400 level) chosen?
The course builds upon and is an extension of the computing and analysis knowledge introduced in CMSC 201 – Computer Science I. Together with CMSC 201, this course will provide the necessary background to proceed to more general, higher-level computing courses (much as CMSC 202 – Computer Science II does for computer science majors).

f) Explain the appropriateness of the recommended course preparation(s) and prerequisite(s).
This course builds directly upon CMSC 201 – Computer Science I. MATH 150 – Calculus I is the prerequisite for CMSC 201 and CMSC 202 and is, therefore, appropriate for this course. Many non-majors are likely to have taken MATH 155 – Applied Calculus rather than MATH 150 (required for CS/CE majors). Either MATH 150 or MATH 155 provides the appropriate math preparation.

g) Explain the reasoning behind the P/F or regular grading method
Students are most likely to take this course using A-F, but on occasion a student could audit or take it P-F.

h) Provide a justification for the repeatability of the course.
This course cannot be repeated for additional credit.

ATTACH COURSE SYLLABUS (mandatory):

CMSC 210: Advanced Computing

Prerequisites:

CMSC 201 with a C or better

Instructor:

Name: TBD
Office: TBD
Office Hours: TBD
Phone: TBD
Email: TBD

Course Description:

This course strengthens and extends the student's programming and problem-solving skills through the use of advanced programming language constructs, pre-defined libraries, and proper software engineering techniques. Topics include program design, debugging, and testing, source code versioning control, use of a software development environment, data formats, web programming, web data extraction, and data visualization. This is the second course for non-computer science, non-computer engineering majors interested in pursuing further study in applied computing.

The goal of this course is to advance the student's computing skills in a manner that they can be applied to a variety of disciplines. Many topics are data-centric and, therefore, suitable to many academic areas such as the psychological, social, and natural sciences. Students receive hands-on experience with a current programming language, development environment, and software applications appropriate to the gathering, manipulation, and reporting of data.

Credits:

Three credits: not repeatable

Learning Outcomes:

At the end of the course, the student will:

- Demonstrate programming skills at a level that can be applied to a variety of problems they will face during careers in their own disciplines/major/field
- Make use of problem-solving skills, especially in the use of computers to solve real-world problems
- Write solutions to real-world problems using a current programming language such as Python
- Use an integrated development environment (IDE) such as PyCharm
- Backup and share source code using software versioning control software such as Git/GitHub
- Understand and implement general data storage, retrieval, and manipulation techniques
- Develop software applications to manipulate and visualize data
- Develop client-side web applications

Grading Criteria:

Type	Points Per	Subtotal
Assignment 1	100	100
Assignments 2 - 7	150	900
Total		**1000**

Grading Scale:

90% - 100%	A
80% - 89%	B
70% – 79%	C
60% - 69%	D
< 60%	F

Readings:

Lutz, Mark (2013). Learning Python. Sebastopol, CA: O'Reilly Media (5th ed). ISBN: 1449355730

Grinberg, Miguel (2018). Flask Web Development: Developing Web Applications with Python (2nd ed). Sebastopol, CA: O'Reilly Media. ISBN: 9781491991732

Chapagain, Anish (2019). Hands-on Web Scraping with Python. Birmingham, UK. Packt Publishing. ISBN: 9781789533392.

Magana, A. and J. Muli (2018). Version Control with Git and GitHub. Birmingham, UK. Packt Publishing. ISBN: 1789808979.

Course Topics:

- Introduction: Setting up your computing environment and programming review
- Advanced programming: Advanced programming language constructs and pre-defined libraries
- Programming environment: Using an integrated development environment (IDE)
- Software engineering: Designing, testing, and debugging programs
- Source code control: Using source code versioning software
- Data formats: Using and implementing text, CSV, JSON data formats
- Software development: Developing client-side web applications, web data extraction, data manipulation and visualization

Academic Integrity:

Academic integrity is an important value at UMBC. By enrolling in this course, each student assumes the responsibilities of an active participant in UMBC's scholarly community in which everyone's academic work and behavior are held to the highest standards of honesty. Cheating, fabrication, plagiarism, and helping others to commit these acts are all forms of academic dishonesty, and they are wrong. Academic misconduct could result in disciplinary action that may include, but is not limited to, suspension or dismissal.

More information can be found at:
https://academicconduct.umbc.edu/

Student Disability Services:

UMBC is committed to eliminating discriminatory obstacles that may disadvantage students based on disability. Services for students with disabilities are provided for all students qualified under the Americans with Disabilities Act (ADA) of 1990, the ADAAA of 2009, and Section 504 of the Rehabilitation Act who request and are eligible for accommodations. The Office of Student Disability Services (SDS) is the UMBC department designated to coordinate accommodations that would allow students to have equal access and inclusion in all courses, programs, and activities at the University.

If you have a documented disability and need to request academic accommodations for access to your courses, please refer to the SDS website at sds.umbc.edu for registration information and to begin the process, or alternatively you may visit the SDS office in the Math/Psychology Building, Room 212. For questions or concerns, you may contact us through email at disAbility@umbc.edu or phone (410) 455-2459.

If you require accommodations for this class, make an appointment to meet with your instructor to discuss your SDS-approved accommodations.

Tentative Schedule:

Date	Topic	Learning Style	Assignments
1/28/2020	Intro + Python Review	Lecture and Live Coding	
1/30/2020	More Python Features	Lecture and Live Coding	
2/4/2020	Installfest (software installation)	Active Learning Exercises	Assignment 1 Released
2/6/2020	Local Development + Git	Lecture and Live Coding	
2/11/2020	Classes and objects	Lecture and Live Coding	
2/13/2020	Using Versioning (Git)	Active Learning Exercises	
2/18/2020	Object-Oriented Programming	Lecture and Live Coding	Assignment 1 Due Assignment 2 Out
2/20/2020	Object-Oriented Practice and IDE Features	Active Learning Exercises	
2/25/2020	Data and Data Formats (JSON, CSV, etc)	Lecture and Live Coding	
2/27/2020	Data Practice and Debugging in IDEs	Active Learning Exercises	
3/3/2020	Building a Website	Lecture and Live Coding	Assignment 2 Due Assignment 3 Out
3/5/2020	Website Building Practice	Active Learning Exercises	
3/10/2020	CSS and Bootstrap	Lecture and Live Coding	
3/12/2020	CSS and Bootstrap Practicce	Active Learning Exercises	
3/17/2020	Spring Break	Spring Break	
3/19/2020	Spring Break	Spring Break	
3/24/2020	Pip and VirtualEnv	Lecture and Live Coding	Assignment 3 Due Assignment 4 Out
3/26/2020	CSS and VirtualEnv Practice	Active Learning Exercises	
3/31/2020	Web Scraping with BS4	Lecture and Live Coding	
4/2/2020	Web Scraping Practice	Active Learning Exercises	
4/7/2020	Web Development with Django	Lecture and Live Coding	Assignment 4 Due Assignment 5 Out
4/9/2020	Django Walkthrough	Active Learning Exercises	
4/14/2020	Databases in Django	Lecture and Live Coding	
4/16/2020	Django Practice	Active Learning Exercises	
4/21/2020	Deployment with Heroku	Lecture and Live Coding	Assignment 5 Due Assignment 6 Out
4/23/2020	Publishing Online	Active Learning Exercises	
4/28/2020	Machine Learning	Lecture and Live Coding	
4/30/2020	Machine Learning Practice	Active Learning Exercises	
5/5/2020	JavaScript	Lecture and Live Coding	Assignment 6 Due Assignment 7 Out
5/7/2020	JavaScript Practice	Active Learning Exercises	
5/12/2020	Angular	Lecture and Live Coding	
5/14/2020	Angular Practice	Active Learning Exercises	
5/19/2020	TBD		Assignment 7 Due

Tentative Schedule