Metropolitan State University
ICS 240 Introduction to Data Structures
Fall 2015

SYLLABUS

Class Hours: Mondays, 6:00 - 9:20
SJH 152
08/24/15 - 12/7/15

Instructor: Sue Fitzgerald, Ph.D.
Metropolitan State University
700 E. Seventh St., NM L116
St. Paul, MN 55106
E-mail: sue.fitzgerald@metrostate.edu
Web site: This face-to-face course will use D2L for assignments
Phone: 651-793-1473
Support staff: 651-793-1471 (Lucy Peterson)

Office Hours (August 24 – December 7):

Mondays	3:00-5:30 pm
Wednesdays	1:00-6:00 pm (except 11/4, 11/25)

- Appointments are encouraged to prevent conflicts, even during office hours.
- I will be available before class and by appointment at other times in addition to office hours.
- I am usually available via email on weekdays unless traveling.
- I check my email occasionally on weekends although response time may be slow.

Required Reading:

- Michael Main, Data Structures & Other Objects Using Java, 4th Edition, Pearson, 2012. ISBN: 0-13-257624-4.
- Sarnath Ramnath and Brahma Dathan, Object-Oriented Analysis and Design, "A Notation for Describing Object-Oriented Systems", Springer, p. 39-47 (posted in D2L).
- Understanding UML Class Diagram Relationships – see http://creately.com/blog/diagrams/understanding-the-relationships-between-classes/

Course description: Students learn intermediate object-oriented design, programming, debugging, testing skills, and algorithms in this course via the study of list, stack, queue and tree abstract data types. Other topics include recursion, hashing, sorting, complexity analysis, and documentation. Design, testing, and complexity analysis are emphasized. Programming intensive.

Prerequisites:

- ICS 141 Programming with Objects or equivalent knowledge of Java AND
- MATH 215 Discrete Mathematics
- This course will be taught using the Java programming language. This course is designed for students who already know the basics of the Java language. Students entering this class should have a basic understanding of looping (*for*, *while*, *do-while*), decision (*if* and *switch*) statements, functions, argument passing and arrays in addition to a rudimentary understanding of objects and classes. Students should understand widgets well enough to write simple user interfaces. A pre-course assessment has been provided to help you see if you are ready for this class.

This course is a required course for the Computer Science major. It is an elective course for Computer Information Technology (CIT) and Computer Application Development (CApp) majors.

Context:
Although the discipline of computer science and its practical applications encompass far more than simply programming, programming is the common language shared by those who practice in the field of computing. As computing professionals, no matter what your job function is, you will be expected to think logically and sequentially. You will be expected to approach problems in a methodical fashion, to experiment and to persist until you find solutions.

This course is designed to give you practice in learning how to approach problems. The most important things that you can learn from this course are how to think about problems abstractly (ignoring non-essential details in order to design solutions), how to deal with complexity by breaking problems down into smaller pieces and how to manage your time.

When you finish this course you will be able to design and implement Java programs at an advanced beginner or intermediate level. The vehicle by which you will meet these objectives is through extensive programming practice.

Competence statement: Knows concepts and approaches to the implementation of lists, stacks and queues well enough to write moderately difficult programs using iteration, recursion, arrays and classes. Understands dynamic memory allocation.

Learning outcomes:

- Can apply elementary data structures such as queues and stacks to solve practical problems including simple simulations.
- Can implement lists using both arrays and dynamic storage.
- Can apply sorting and searching algorithms.
- Understands recursive solutions and can use recursion to solve problems such sorting and searching.
- Can analyze the complexity of algorithms.
- Can store and search data using the basics of more advanced data structures such as trees.

Evaluation: Your evaluation will be based primarily on the following:

- Problem solving skills: Can predict the outcome of sample code and can solve simple programming problems on tests and homeworks.
- Program design and implementation: Is able to design and write simple application programs in Java using concepts covered in class. Demonstrated through programming assignments.
- Programming style: Can produce readable, well-documented code that conforms to style conventions, to be evaluated by computer programming assignments.
- Abstract thinking: Can explain data structures concepts.
- Responsibility: Attends class, turns in work and meets deadlines.

Points are roughly allocated to different learning activities as indicated in the table below:

Description of Item	Number of Items	Maximum Points per Item	Total Maximum Possible Points
Tests	2	100	200
Final Exam (comprehensive)	1	200	200
Programming Assignments	6	50	300
Class participation/ homework	~10	~10	~100
Total			Approx. 800

*There will be some variation in the number of assignments and the points or weight awarded for each assignment may vary. The total shown above is an approximation of the final number of possible points.

Letter Grade: Your letter grade will be determined based on the percentage of possible points that you earn during the semester. The following table relates the percentage to a letter grade:

Percentage	Grade
94 – 100	A
90 – 93	A-
87 – 89	B+
84 – 86	B
80 – 83	B-
77 – 79	C+
74 – 76	C
70 – 73	C-
60 – 69	D
Below 60 %	F

If you have selected the S/N grading option, then to receive a satisfactory rating (S) you must earn at least 70% of the possible points. If you wish to elect S/N grading, you must inform the registrar before the second class meeting. Bear in mind that you must earn a C- or better (or an S) in order for a course to fulfill a requirement or elective in your major.

Incompletes: From time to time I am asked to consider assigning a grade of incomplete. A grade of incomplete may be considered if the person requesting has successfully completed at least two thirds of the class and is a student in good standing in the class. "Good standing" means that the requester is earning a minimum of a B grade and has attended class regularly. I reserve the right to say no to any request for an incomplete without justifying my position.

Time Requirements: For many students, computer programming is quite time-consuming. I know of no way to learn to program without extensive practice, so there will be both reading and programming work to be done outside the class period. The best way to become a good programmer is to practice at every opportunity.

The amount of time required to prepare for this class each week will vary from individual to individual. It takes time to complete the required reading assignments, problem sets and especially the programming assignments. Also, you will need to study for the tests. Some people estimate that it may take 3-4 hours of preparation outside of class for each hour spent in class (10-13 hours per week). So you need to be prepared to spend at least that amount of time each week, and it may take more time than that. Of course, it may actually take less time than this for you.

Attendance will be taken every night, sometimes more than once per class session. You are expected to attend class and will not be eligible for points awarded for in-class exercises if you do not attend.

Software: All programming assignments are to be done in Java. Note that for grading purposes, I have to be able to compile and execute your program. If your compiler causes grading problems, then I will ask you to use what is available in the lab. We are using Eclipse Version: Kepler Service Release 1, Build id: 20130919-0819 and Java version jdk1.8.0_51 and jre1.8.0_51. These versions are installed in all Metro State labs and are available for free download. Most students in this class will use the Eclipse IDE.

Homework Problem Sets: There will be a number of in class exercises and there may be homework problems sets as well. If assigned, homework answers will be due at the beginning of the class period. Because the answers will be given in class, no late homeworks will be accepted even if you are not present in class. Credit will not be given for "in class exercises" if you are not in class. Homework should be submitted via D2L.

Programming Assignments: There are 6 planned programming assignments. They are due before 11:59 PM of the assigned due date. It is essential that you keep up in this class. To encourage you to stay on track, a 10% extra credit early submission bonus will be available for some programs. Programs submitted at least 24 hours before midnight on the due date are eligible for the early submission bonus. However, if the program is not turned in by the due day, a 10% per day late penalty will be applied to late submissions. No program will be accepted more than 4 days late (Friday after the due date). **DO NOT MISS CLASS IN ORDER TO WORK ON A PROGRAM**. This can only make you fall further behind.

Most programs will be worth 50 points each. The grading of the programs will take into account programming style, design documentation and correctness. Point values associated with the factors will be indicated in the assignment. Generally, correctness will be weighed most heavily.

Programming projects containing Java source code must be zipped and turned in via D2L. See the Supplementary materials for a description of how to export your Java files for upload to D2L. Do not email your assignments to me. They will be rejected.

Electronically submitted documents or homeworks must be in one of the following formats: pdf, txt, rtf, Word 2003, 2007, 2010, 2013, or jpg. If I cannot read it, I cannot grade it.

No work will be accepted after the last scheduled class period.

It is a good practice to keep a backup copy of your work.

No Makeup Tests: No makeup tests will be offered. If you miss a test, the next exam will be doubly counted in order to replace the missed test. To exercise this option, you must inform me via email of your valid reason for missing the test, the earlier the better. Requests made more than one day following the test may be declined.

I strongly believe that all students should take the final exam at the designated time. This helps to ensure that all students are tested in a uniform manner. If you do convince me to write a separate final exam for you, it may be more difficult than the exam given to the rest of the class. I reserve the right to refuse to give you a makeup final exam.

Missing Class: In this course you will be practicing problem solving skills and programming concepts and techniques. I strongly recommend that you attend all classes so that you can observe and practice the process of solving computer-based problems from start to finish. If you know that you will miss more than one class this term, I encourage you to drop the course now and take it during a term when you can attend all class periods.

If for any reason you need to miss a class, it is courteous to let me know beforehand if possible. Materials can be downloaded from D2L. Please make arrangements to borrow or copy a classmate's notes from that session as I will not give recap sessions for those who miss class. Assignment due dates and late assignment penalties will apply whether you attend class or not. I suggest that you exchange phone numbers and/or email addresses with several others in the class the first night. The Saturday group tutoring sessions are another good way to connect with your classmates.

Expectations: Sometimes difficulties arise when the expectations of students and teachers are different. Here is a brief summary of what I expect from you AND a summary of what you can expect from me.

What you can expect from me:

- I will share my technical knowledge with you as effectively as I know how.
- I will grade your work and return it to you via email before the next class meeting (within one week). Occasionally I may take longer to return your work although I will try not to.
- I will provide you with written feedback on the quality of your work. If you do not understand why you were assigned a particular grade or if my comments are not clear, I will explain more thoroughly if you ask me to.
- I will show up for class or provide a substitute teacher. I will start and end class on time.
- I will return your phone calls and emails. Although student emails and calls are a priority for me, I may take more than one business day to respond.
- I will listen to you respectfully. I will answer your questions respectfully.

What I expect from you:

- I expect you to read the syllabus and to know the class policies outlined there.
- I expect you to seek help if you are having difficulty with your course work. I expect you to talk to me if you are having problems. If a stressful work or home situation arises which is affecting your ability to perform well, please talk to me as soon as possible.
- If you do not understand why you received a particular grade, I expect you to approach me and ask questions. I expect you to treat me with respect even if you disagree with the grade you were assigned.
- I expect you to show up for class on time and stay for the entire session except when you have an unavoidable conflict. If you must miss class, assignments and most handouts can be obtained from D2L.
- I expect you to turn in your assignments. I expect them to be submitted in a timely fashion.
- I expect you to come to class prepared to discuss all homework assignments on the day they are due.
- I expect you to participate in class.
- I expect you to know the class late policy. (A penalty of 10% per day will be assessed for late programs. No programming assignment will be accepted more than four days past its assigned due date. Written homework assignments must be turned in on time – there are no late penalties on written homework because late homework will not be accepted.)
- I expect you to turn in individual and original work. Please be respectful of copyrights and document your sources appropriately. This means that what you turn in must be your own work and it cannot be work previously done for some other class. Nor is it acceptable to download solutions from the web. If you copy the words or code snippets of another, you must acknowledge the source. Otherwise, you have committed plagiarism. If you turn in work done by someone else you may be assigned an F in this course. Repeated instances of academic dishonesty can result in expulsion from the University.
- I expect you to say no if another student asks you for a copy of your work. The penalty for permitting another student to copy your work is the same as if you yourself had turned in work that was not individual and original.
- I recommend that you complete the reading assignments before you come to class, particularly if you are having difficulty understanding the material.
- I expect you to turn off your cell phone during class. If your cell phone must remain on, please turn your phone to vibrate.
- I expect you to refrain from reading email, texting, surfing the web, playing computer games or doing your homework assignments during lecture.
- I expect you to remain in your seats during lecture except for emergencies or religious observances. Regularly scheduled breaks will be provided.

Respect: Metro State is privileged to serve students from many different nations, racial, ethnic and religious backgrounds. Students, staff and faculty practice a variety of lifestyles and come from many walks of life. We expect our classrooms to be safe havens where the opinions, practices and beliefs of others are treated respectfully. If you feel that you are not being treated appropriately by others in the class, I ask that you bring this to my attention so that the issues can be addressed. If I offend you, I ask that you approach me to share your concern so that we can learn from each other how to live together with respect and honor.

If you do not feel comfortable sharing your concern directly with me, contact the Information and Computer Sciences Department chairman, Mike Stein, at 651-793-1476 or michael.stein@metrostate.edu.

Learning Disabilities: If you have a documented learning disability, or if you suspect you have a learning disability that may have an impact on your opportunity to succeed in this course, please talk to me before the second class so we can explore ways to reasonably accommodate your learning style. Accommodations requested for you by Disability Services will be honored. To reach Disability Services, call 651-793-1549 or email disability.services@metrostate.edu. Note that I must receive adequate notice in order to honor requests.

Academic Honesty:
Academic integrity is a fundamental element of the learning process. Only by assessing your own original work can I determine whether you've learned and met the educational goals I have developed for you in this course. For that reason, we take academic integrity very seriously in our learning community. It is your responsibility as a student to read and understand Metropolitan State's *Academic Integrity Policy and Procedures*, which can be found here:

Procedure: http://www.metrostate.edu/applications/drep/files/Procedure%20219.pdf
Policy: http://www.metrostate.edu/applications/drep/files/Policy%202190.pdf

More specifically, all programs and documentation must be individual and original work unless the assignment specifies otherwise. If your program closely resembles someone else's, I will call it to your attention. Identical programs will be assigned 0 points. Two or more programs which closely resemble each other will be treated as one program and the possible points for one program will be divided among the students who submitted those programs. Repeated instances of similar or identical programs will result in a grade of F for the class. A pattern of academic dishonesty may result in expulsion.

To avoid having your programming assignments look similar to someone else's, you should not look at another person's design until you have written down your design; you should not look at someone else's code until you have written down your code. This rule of thumb also applies to tutoring. The tutor should not assist you until you have firsts attempted to solve the problem on your own.

If we have a group assignment, then the work turned in can be based on the contributions of the other group members. If such an assignment is made, it will be clearly indicated on the written assignment.

However, it is OK to ask for help. Learning to program can be an intensely frustrating experience unless you learn when to ask for help. In general, if you spend more than 15 minutes absolutely stuck on a problem, you should ask someone else to give you a hint. However, you should *never* copy someone else's solution.

University Policy on Academic Progress: The university's academic progress policy may affect students who withdraw from classes. Be aware that a W (withdraw) is different from a drop. A drop occurs at the very beginning of the term (no later than Sept 1 for this course), while a withdraw occurs

after the first week (between Sept 2 and Nov 21 this term). Withdrawing from this class may put you at risk for academic probation. If you have questions about your situation, contact your academic advisor as soon as possible.

University Policy on Attendance: Students who attend neither the first nor second night of class will be dropped from enrollment.

Hot tips: Here a few suggestions for success.

1. Start programs as soon as they are assigned. Attempt to complete them one week early. This way you will be sure to find any problem areas in time to bring your questions to class before the program is due. NEVER wait until the night before a program is due to start it.
2. Turn in every assignment on time – try for the early submission bonus!
3. Scan through the assigned reading before coming to class.
4. Find a study partner or group of partners. Exchange phone numbers and email addresses. Arrange times when the people in your group plan to work in the lab. Although I expect each of you to work independently and do your own work, the others in your group will be able to help you with syntax (compile) and logic errors. Besides, it's more fun.
5. Don't stay stuck!! If you cannot figure out a problem after 15 minutes of thought, ask for help. Don't waste time on syntax errors.
6. I hope to arrange for Saturday morning drop in Java tutoring in the lab. Plan to attend the Saturday morning group tutoring sessions. Work on your assignments then if you do not need help.
7. Plan ahead. A good design will save you hours of coding and debugging. Do not sit down at the computer before you have thoroughly thought out the solution to the problem.

Smoke Free Campus
As of May 1, 2014, Metropolitan State's Saint Paul Campus is smoke-free and tobacco-free. Smoking, the use of smokeless tobacco products, e-cigarettes and unregulated products are not allowed on Metropolitan State-owned properties. With this new policy, the university joins more than 1,000 colleges and universities in the nation that classify themselves as 100 percent smoke-free. The policy, FAQs and cessation resources can be found at http://www.metrostate.edu/breathefree. Questions regarding the policy can be directed to breathefree@metrostate.edu or Dan Hambrock, associate vice president of facilities, at 651-793-1712.

Metropolitan State University
ICS 240 Introduction to Data Structures
Fall 2015

Schedule

Week	Dates	Topics	Reading	Assignments
1	Aug 24	Syllabus Review of classes, UML class diagrams, UML structure diagrams, comparison of objects, arrays of objects, searching, runtime analysis	Syllabus Chapters 1 and 2 Chapter 11 (p. 568-570 only) Object-Oriented Analysis and Design, p. 39-43 (see D2L for handout) Appendix C Throwing and Catching Java Exceptions (review topic)	
	Aug 29	Java Review	NM L103 9:00 – 11:30 am	Java Review
	Aug 30	LAST DAY to take pre-course assessment for homework credit		Pre-course assessment
	Aug 30	LAST DAY TO DROP WITH REFUND		
2	Aug 31	Collection Classes, the clone method Javadocs	Chapter 3 "Understanding UML Class Diagram Relationships" – found at http://creately.com/blog/diagrams/understanding-the-relationships-between-classes/ Appendix H	
	Sept 7	NO CLASS – LABOR DAY		
3	Sept 14	Generics Simple sorting algorithms, the Comparable interface, iterators	Chapter 5 (p. 251-282, 286-299, 309 only) Section 12.1 Appendix G Further Big-O Notation	Program #1 due
4	Sept 21	Stacks (guest speaker)	Chapter 6 (p. 316-340 & p. 345-355)	
5	Sept 28	Queues	Chapter 7 (p. 360-393)	Program #2 due
6	Oct 5	Linked lists	Chapter 4	Test #1

7	Oct 12	Linked lists	Chapter 4 (continued) Chapter 5 (p. 283-285 only) Appendix E	
8	Oct 19	Linked implementations of stacks & queues	(p. 341-344 & p. 393-399)	Program #3 due
9	Oct 26	Recursion Binary search	Chapter 8 Chapter 11 (p. 571-580 only)	
10	Nov 2	Recursion	Chapter 8 Section 11.1	Program #4 due
11	Nov 9	Trees; Binary Search Trees	Chapter 9	Test #2 Program #5 design due
12	Nov 16	Binary Search Trees; B-Trees	Chapters 9, 10	Program #5 due
13	Nov 23	Hashing and Recursive Sorting Algorithms	Sections 11.2, 12.2	
	Nov 23	LAST DAY TO WITHDRAW		
14	Nov 30	Catch up & review for final exam		Program #6 due
15	Dec 7	Final Exam		Final
	Dec 15	Alternative day for finals cancelled due to bad weather		

* Adjustments to the schedule and assignments will be made as needed.

Programs

1. Arrays of objects
2. Sorting, searching
3. Simulation, queues
4. Linked lists
5. Recursion
6. Trees