CS 101 Fundamentals of Computer Science

Course Syllabus

Prof. Dean Zeller & Dr. Joe Villani

York College of Pennsylvania

Fall 2025

Professor Dean Zeller
Email: dzeller@ycp.edu
Cell: 970-576-4966
Office: 136 KEC, 717-815-6651
Office Hours: by appointment or
any time I am in the office with door open.

Dr. Joseph Villani
Email: jvillani@ycp.edu
Cell: 717-881-9506; text me first so I know who is calling.
Office: 110 KEC
Office Hours: by appointment

Our goal is to make this class a wonderful learning experience that is enjoyable and relevant to your future. Please let us know if there are any questions or concerns about the class, and it will be immediately addressed. Be yourself and express your honest ideas.

Table of Contents

Course Overview

This course introduces the fundamental techniques of algorithm design and program construction using procedural constructs. Topics will include problem analysis; algorithm design; and implementation and debugging strategies using good programming practices. The course will cover basic data structures including variables, arrays, strings, records, and pointers; and control structures including decisions, iterations and functions. The course will focus on applications from computer science and engineering using C/C++.

There are no prerequisites for this course. There is no required textbook.

Course Structure and Expectations

Class meetings will be primarily a discussion of various programming concepts illustrated through sample code in C. It is important to come to class prepared to ask questions related to the topic and/or work on the lab exercises which are designed to reinforce the concepts from the lecture notes. There will be a series of homework programming assignments to be completed individually.

Each class will be structured as follows,

- 20 minutes : Team problem exercises discussed prior class
- 20 minutes : Lectures covering new material, and
- 35 minutes : Lab and assignment work, with access to professor for questions and debugging.

All the programming assignments are cross platform using C; see the class resource page for instructions on installing the necessary components if you wish to use your own machine.

We will be covering a significant amount of material in the course at a rapid pace, so it is imperative that you keep up by participating in the class meetings. Class attendance is mandatory where the instructors will strive to for classes to be interactive, collaborative, and exploratory.

Learning Outcomes

The basic objectives of this course are to provide the student with enough programming tools and methods to feel comfortable writing C / C++ programs that solve problems encountered in computer science and engineering. The following topics will tentatively be covered:

- Be able to create, compile, and run C programs using the Gnu C/C++ compiler
- Be able to declare variables of various types and write programs that do basic arithmetic operations
- Understand and apply control structures such as conditionals and loops
- Understand and use arrays
- Understand and be able to write functions
- Understand and be able to define and use structure types
- Understand and be able to use pointers
- Use top-down design to decompose complex problems into simpler problems

Teaching Methods

The instructors take a reflective approach to teaching. We continually reflect on and improve on our teaching methods. To do this, we ask for your feedback and adapt our teaching. During week 6 of the course we will provide a short, three-question anonymous survey to assist us in adapting our teaching approach to best suit the learning outcomes of our students.

Grading Criteria & Evaluation

Grade Component	Points	Weighting %
Class Attendance and Participation (15)	150	10%
Fifteen (15) Labs	150	10%
Five (5) Homework Assignments (programming with report and video components)	300	20%
Exams (3 midterms and 1 *optional* cumulative final exam, lowest score dropped)	900	60%
Total	1500	100%

Labs

- Students will work on labs during class as a critical learning component that reinforces the lectures.
- Labs are to be completed in class or during tutoring hours.
- To receive credit for a lab, you must get a sign-off on the completed program during class; full credit if you get this sign off by the end of the *next* week after the day we work on the lab, 80% of full credit if you get the sign-off before the next exam, and **no credit** after the exam.

Class Attendance and Participation

- Do not miss class! If you do not attend class, show significant effort on the lab assignment by the end of the day, or if you are seen doing non-lab activities before finishing a lab, you will be considered absent.
- Missing 20% of the classes, or non-participation in class will result in a reduction of your grade by 1 letter grade.
- It is the student's responsibility to get notes, announcements, and homework assignments from other students or the instructor if a class is missed.
- It is strongly recommended that you look over the material for each day before coming to class.
- A presentation could include a student chosen at random and asked to briefly demonstrate the assigned code (2 minutes), then asked to demonstrate minor changes (5 minutes).

Homework Assignments

You must demonstrate significant effort on all homework assignments in order to receive a passing grade the course regardless of your overall course average. The very important outcome of using top-down design to decompose complex problems into simpler problems cannot be assessed based on exams alone. This will be demonstrated via homework assignments. Failure to demonstrate this outcome on homework assignments will be grounds for failure even with a high exam average.

Exams

There will be three midterm exams, and an *optional* comprehensive final which may replace your lowest midterm exam score. Exams will be closed-book, closed-notes and will include a programming component. A single page language reference sheet will be provided for exams. No make-up exams will be given without approval of the instructor prior to class unless proof of extreme emergency or illness is provided.

Grading Turnaround

It is planned that all instructor grading will be completed over the weekend. Gradeslips will capture an accurate snapshot of grades completed. Alert the instructor if there was a submission not graded for two weeks, as there could have been a break in the submission or grading process.

Grading Scale

The grading scale for the course is given in the table to the right. Please note that this grading scale represents the most stringent criteria the students will have to meet to get a particular grade. Each instructor reserves the right to reduce (i.e. curve) the grade scale based on the final course performance.

Grade	Range
4.0 (A)	≥ 90 and < 100
3.5 (B+)	≥ 87 and < 90
3.0 (B)	≥ 80 and < 87
2.5 (C+)	≥ 77 and < 80
2.0 (C)	≥ 70 and < 77
1 (D)	≥ 60 and < 70
0 (F)	< 60

Graded Work Policy

The following policy pertains specifically to all graded work in this course:

- All graded (individual) assignments are to be completed individually. We encourage you to discuss high level concepts with other students, but any work you submit must be yours alone.
- Direct copying of solutions or work from other students, web sites, or other sources is absolutely forbidden under any circumstances.
- Any sources (books, websites, articles, fellow students, etc.), except for lecture notes, that you consult in completing an assignment must be properly acknowledged.
- Exams must be completed individually using only the resources provided by the instructor.
- Discussing programming assignments with other students is acceptable and encouraged only for high level conceptual design. You must write and understand all of the code you include in your program.
- Verbatim copying of someone else's code (including electronic transmission or using code from websites) is STRICTLY FORBIDDEN under any circumstances.
- You must actively contribute to the process of understanding and solving the problem posed in the homework assignment. Tutoring will be available if you need assistance in writing code.
- When you collaborate with another student, you must add comments to the code documentation you turn in stating who you worked with, and what the nature of your collaboration was
    - For example, if you worked with Alice Smith, you might add a comment as follows:
    - *// CREDITS*
    - *// I worked with Alice Smith. We worked together on constructing the*
    - *// loop in the compute_temperatures function.*
- When you collaborate with other student(s), you must make sure that everyone involved in the collaboration properly acknowledges everyone else who collaborated. In the example above, Alice would need to acknowledge working with you when Alice turns in the assignment.

Violations of the policy, such as failing to cite collaborators or electronic copying of code, will be considered a violation of academic integrity subject to sanctions.

Regardless of your overall course average, to pass this course with a 2.0 (C) grade, you must meet the following criteria.

1) Complete 4 homework assignments with a grade of 70%.
2) Receive sign-off on 10 required labs.
3) Participate in 80% of the classes.
4) Receive a grade of 70% or higher on at least one exam, *and* have an average of 60% or higher on your three highest exam scores.

Course Guidelines & Expectations

Late policy

Except for exams, students will have flexibility in turning assignments within the requested completion date. It is recommended that students resubmit their assignments as they receive feedback from the instructor. Current labs and assignments must be completed in before the next exam to receive credit.

Professionalism

We expect you to conduct yourself as a professional in this course. Professionalism includes:

- Respect for and courteous interaction with peers, faculty, and facilities;
- Integrity, which includes at its core honesty, responsibility and accountability for one's own actions;
- Sensitivity and appreciation for diverse cultures, backgrounds, and life experiences;
- Constructive evaluation, which means that criticism is offered and accepted in a productive manner;
- Self-reflection and identification of one's own strengths and weaknesses;
- Responsibility for one's own education and learning;
- An attitude that fosters professional behavior in colleagues and peers;
- Punctuality at meetings and class sessions;
- Attentive behavior during class sessions, avoiding personal or social use of any electronic devices;
- Acknowledgement of the Kinsley Engineering Center as a professional workplace, and treatment of this facility as a business or office space, not as an informal space.

The instructors reserve the right to enforce this code through the York College Code of Student Conduct.

Overall Behavior

This class follows the York College regulations on student behavior and conduct. It is expected that students behave maturely, professionally, and will be respectful to the instructor and fellow students. Please do not be disruptive to the teacher or to the other students. The instructor reserves the right to remove any disruptive students from class.

Use of Appropriate Language

Please use language appropriate for everyone, realizing that other people in the classroom may be within earshot of the conversation. This applies to all forms of communication (verbal, online, written on board, etc…).

Face-to-Face Communication

Please make every attempt to communicate with your instructor face-to-face. In-person communication is, by far, the most effective method, and has the least amount of opportunity for miscommunication or misunderstanding. While simple tasks may be completed over phone, text, or email, anything complicated or potentially ambiguous should be left until it can be handled face-to-face. The course instructor will have regular office hours to encourage in-person communication.

No Zoom Requests

Attendance through Zoom is not acceptable and will not be counted or prepared for. Students are to attend class and office hours in person.

Getting Help

Students are taught to be self-reliant. Part of this trait is to know when to ask for help, and how to get the greatest benefit. When requesting for help in the industry, programmers must understand that while others are generally willing to help, they have their own schedules to keep. It is considered polite to make certain the problem has been well-investigated beforehand, so as to not waste time. Be ready to discuss the problem, what has been tried in the past, and the previous efforts to solve the problem. This applies to seeking help from tutors as well.

Virtual Meeting Days (previously known as "Snow Days")

The concept of a "snow-day" is being altered within the educational norm. At York College, snow-days were replaced with "virtual meeting days," where instruction and requirements should occur online during classtime instead of in class. Having said this, the instructor will never hold required Zoom meetings in lieu of class. If there is an unexpected snow-day, check Canvas and email for any announcements or alternate requirements, but there will not be a Zoom meeting during class time, as the Internet may not even be available for students without power. Students are free to use the canceled class-time as they see fit for a potential emergency situation.

Student Concerns

For any concerns about the class, it is requested that student contact the professor directly to address the problem. In most cases, a mutually satisfactory solution can be reached. Students unsatisfied with the solution presented by the professor can take the case to the department chair, Dr. Kala Meah, who will address the issue further.

Privacy and FERPA

FERPA is the Family Educational Rights and Privacy Act, a federal law that protects the privacy of student education records. To summarize, grades and personal information for a student are completely confidential between the instructor and the student. There are no other parties that have access to student information, including parents, employers, or acquaintances. Students may complete a FERPA form to allow specific people access to their records. Note that this form only allows access to stored information, but does not allow for direct interaction. Even with a FERPA form, any discussion of grades can only happen between instructor and student.

Modifications to Syllabus

The instructor reserves the right to modify the syllabus and assessment criteria in response to student needs, developments, and unforeseen events. The instructors take a student-centered approach toward teaching and may augment the class structure or offer other learning experiences, such as optional encore lectures, to overcoming difficulties with the material.

Inclusivity Statement

York College of Pennsylvania aims to provide access to high-quality education. To reach our mission, we value bringing different voices together to share their strengths. In this diverse environment, when we create a sense of belonging and equity, we enrich the learning experience and better serve our local and global communities.

As such, York College:

- Welcomes and recognizes diversity in all of its forms and expressions.
- Encourages civil conversations across our campus to promote understanding.
- Stands up against prejudice and injustice in our college community.
- Advocates for the well-being and dignity of our community members.
- Enables marginalized identities to be their authentic selves.

Accommodations Statement

York College of Pennsylvania offers a variety of academic accommodations to students with documented disabilities to ensure their success. To request accommodations, please contact Student Accessibility Services at (717) 815.1717 or sas@ycp.edu. Student Accessibility Services will discuss the confidential process of requesting accessibility services and establish the accommodations for which the student is eligible.

Technical Policy: Use of Personal Technology in the Classroom

While York College recognizes students' need for educational and emergency-related technological devices such as laptops, mobile devices, cellular phones, etc., using them unethically or recreationally during class time is never appropriate. The college recognizes and supports faculty members' authority to regulate in their classrooms student use of all electronic devices.

Communication Standards

York College recognizes the importance of effective communication in all disciplines and careers. Therefore, students are expected to competently analyze, synthesize, organize, and articulate course material in papers, examinations and presentations. In addition, students should know and use communication skills current to their field of study, recognize the need for revision as part of their writing process, and employ standard conventions of English usage in both writing and speaking. Students may be asked to further revise assignments that do not demonstrate effective use of these communication skills.

Students who want any assistance with oral, written, and/or visual communication are highly encouraged to contact the Writing Center. The Writing Center is located on the ground floor of the Humanities Building 019.

Student Accessibility Services Statement

York College and Student Accessibility Services (SAS) is dedicated to providing an inclusive campus environment and quality learning experience that is equally accessible to all students. York College will provide reasonable accommodations and support services to any eligible student with a disability. The goal of Student Accessibility Services is to help students access education, demonstrate their knowledge, and meet their academic potential. Students can request accommodations by completing and submitting the SAS online application, along with documentation related to their disability. The application, documentation requirements, and additional information can be found at www.ycp.edu/sas. The student will then be invited to meet with SAS staff to discuss appropriate accommodations and plan a strategy for student success. York College encourages any student that feels they may qualify for accommodations to contact Student Accessibility Services to set up an appointment.

Disability Support Services (Adopted 2015)

In accordance with the provisions of Section 504 of the Rehabilitation Act of 1973, and the Americans with Disabilities Act of 1990, York College and its faculty are obliged to make reasonable classroom and physical accommodations for students with disabilities. If you are a student with a disability in need of classroom accommodations and have not already registered with Julie Rasmuson, Student Accessibility Services, please contact her at 815-1443 or jrasmuson@ycp.edu to discuss policies and procedures related to disability services and to establish the accommodations for which you are eligible.

The Accommodation Memo: Students with disabilities are responsible for providing the Director of Student Accessibility Services (SAS) with proper documentation to determine eligibility for accommodations, and meeting with the Director of SAS to obtain their accommodation memo. The student is then responsible for showing the memo to each of their professors where accommodations are needed and discussing any questions or concerns. Additional information for faculty and students regarding Accessibility Services and Accommodations is available at: Student Accessibility Support Services

Academic Integrity Policy

The YCP Academic Integrity Polity is summarized below. For the full policy, go to the Academic Standards section of the current Course Catalog (https://www.ycp.edu/about-us/offices-and-departments/registrar/catalogs/). York College of Pennsylvania, as an institution of higher education, serves to promote and sustain the creation, acquisition, and dissemination of knowledge. In order to fulfill this purpose, an environment of integrity, dependability and honesty must be maintained by all members of the York College community. Without a foundation based on intellectual honesty and integrity, the very ability to uphold the academic endeavors that York College strives to pursue is inhibited.

Academic integrity involves two fundamental expectations:

- Anything you turn in as your own work is, in fact, your own work and your own words, completed without assistance, unless your instructor has given explicit permission otherwise.
- Anything you turn in is truthful. Lab data were generated in the lab (and not made up), hours worked for an internship or coop were actually worked, etc.

The Academic Integrity Policy includes a non-exhaustive list of activities that are prohibited. Some of the commonly encountered prohibited activities include:

- Plagiarism (passing someone else's words or ideas off as one's own without proper attribution).
- Getting assistance from other students on non-collaborative assignments. You are permitted (and encouraged) to get assistance from your instructor and the Academic Success Center.
- Sharing papers, exams, homework assignments, etc. with other students (even if it wasn't your intent to cheat).
- Ghostwriting (getting someone else to write a paper, whether it is a friend, an essay mill, or a generative AI tool).
- Using unauthorized assistance on exams (e.g., cheat sheets, websites, publisher test banks, other students).
- Buying/sourcing assignment answers from other people (whether it is other students, a website like Chegg, or other online sources).
- Turning in papers/assignments completed in other classes.

This is not a complete list of prohibited activities. Check out the policy in the catalog for a more comprehensive list. The onus is on you, the student, to verify that any exceptions are allowed in that class by that instructor. Instructors have full discretion to assign a sanction up to and including a grade of 0 in the class for violations of the policy. Violations will be reported to the Associate Provost of Student Success as outlined in the policy. You cannot withdraw from a class if you have been charged with an academic integrity violation.

If at any point you are unsure whether something is allowed under the academic integrity policy, please ask your instructor!

Artificial Intelligence Use Policy

Artificial intelligence (AI) tools (such as ChatGPT, Copilot, Gemini, etc.) are powerful resources. However, in this class, the primary goal is for you to learn to think like a computer scientist and develop your own problem-solving and programming skills.

As part of the assessment process, students may be asked to explain their code and know how to make requested to explain their code and know how to make requested modifications. Inability to explain a piece of code written can be used as evidence in improper AI usage.

To that end, AI use is restricted as follows:

You may use AI to …

- look up factual information (e.g., formulas, unit conversions, mathematical constants).
- Examples, to confirm that the formula for converting Fahrenheit to Celsius, or checking the relationship between degrees and radians.
- Understand concepts, strategies, or approaches
- Provide examples for learning purposes
- Write initial drafts of paragraphs in reports
- Assist with writing and formatting documentation
- Debugging or optimizing code

You may NOT use AI to

- Write entire programs by using the assignment as a prompt
- Replace human learning

All programming work must be entirely your own.

The programming assignments are designed to help you practice logic, problem-solving, and coding skills. Using AI to complete them defeats the learning purpose and violates course integrity.

Any submission containing unethically generated code using AI will be treated as academic dishonesty and subject to disciplinary action according to university policy.

We want you to succeed in this course and come away with real programming skills. These are the skills you'll need in later courses and in your careers. To do that, you need to practice coding yourself. For that reason, AI tools cannot be used to write, debug, or explain code on any assignments. All of your programming work must be your own.

Assignment Guidelines & Expectations

In addition to the design and programming aspects of assignments, students will need to create an assignment report. A report template will be provided to get students started. The assignment report will be structured into the following sections.

- Header
  - Student name
  - Assignment number and title
  - CS 101 Fall 2025, Fundamentals of Computer Science
  - Professor Name
  - Date submitted
- Problem Description
- Input
- Output
- Strategy
- Control Flow Sketch
- Similar Problems
- Results
- References
- Self-Evaluation
- Appendix: Full Code
- Appendix: Full Output

Reports

It is not sufficient to simply turn in a program as an assignment. Students must learn the value and the ease of organizing a full report of their work. Once students get the hang of it, writing a complete report is not difficult. It is fully acceptable to use AI to help generate content and formatting of a report, but students are fully responsible for what is submitted.

Report and Video Resubmissions

In the interest of full and detailed learning, students may resubmit reports and videos for full credit until the end of the semester. The underlying spirit of resubmissions is to have at least one level of teacher edits before the video is recorded.

Report First Draft (10 points)

A report first draft is completed directly after the program is believed to be complete. The purpose of the first draft is to collect all relevant information and write the first version of each section. The idea is to have a document that is ready to be reviewed and edited, but without the necessity of reading the entire document in detail. First drafts are submitted via Canvas, and may be of any reasonable length, and are worth 10 points. The first draft will be quickly viewed by the instructor to check for the most commonly missed elements, including:

- Proper report header information
- Includes appropriate sections in the correct order (content not yet necessary)
- Full documented code included in appendix
- Full program output included in appendix

Report Final Draft (20 points, resubmittable)

Once the first draft has been graded, the student is to create a final draft with all of the specified requirements. Any issues will be reported, with points deducted. Students are free to resubmit reports to fix any issues. Problems and deficiencies will be labeled as either minor or major issues. Once the comments are collected, a grade will be given. Report grades will be assigned as follows out of 20 points:

0	No submission
1	File was submitted, but it could not be read, played, or executed
5	Multiple major issues, or more than five issues combined
10	One major issue, and up to three minor issues
15	No major issues, up to three minor issues
20	One minor issue is acceptable for full credit
21-22	No issues with grading, assignment shows significant extra content and effort

Video Demonstration (20 points, resubmittable)

To illustrate that students can explain the work submitted, video demonstrations are completed. The video component is where the student truly demonstrates that the project was completed correctly. Do not start the video until the final report has been returned and the resubmission changes are complete. Make the video referring to the updated report and any other necessary content. The video grading scheme is similar to the report, referring to minor and major issues. If there are any remaining errors in the report portrayed in the video, points will be deducted for the same mistake, as it should have been fixed before making the video. All videos may be resubmitted until the end of the semester. Videos should be about 5 minutes in length and follow a certain pattern of presentation. Students are responsible for the visuals, audio, and readability of content, including mistakes, so they are to watch the videos to make sure they are of sufficient quality and contain the correct information. The video sections are as follows:

a) Introduction (30 seconds)
   Start the video showing the documentation of the main file. Introduce yourself and briefly describe the problem solved.
b) Code Explanation (90-180 seconds)
   Scroll through the documented code, describing its purpose and what it is doing. Time is limited, so do not go into too much detail.
c) Execution Demonstration (90-180 seconds)
   Execute the code enough times to demonstrate all of the required features of the program. Start with the shorter cases, such as error-checking. If a feature is not demonstrated, it will be assumed that it was not implemented.
d) Conclusion (30 seconds)
   Indicate to the viewer that the video is complete. Comment on what was learned in completing the assignment. Students are encouraged to provide a non-offensive joke (relevant or irrelevant) to conclude the video, but it is not required. (Example: "And remember, you can tune a piano, but you cannot tuna fish.")

Printouts

For ease of communication, the instructor must have a printout of any full reports submitted. Students that submit their assignment to Canvas by Wednesday do not need to make a printout, as the instructor can print it using the campus print service. Those that do not submit it by Wednesday will have until Friday to supply the instructor with a printout for the same credit.

Minor Issues

Assignment feedback could have comments on any issues with the submission, categorized as minor and major. Minor issues include:

- Misspellings, grammar, or messy border alignment in documentation
- Output or report formatted incorrectly
- Minor mathematical or logic errors that do not affect the outcome
- Minor tabbing and code layout issues
- Minor error in output correctness
- Video too short or too long (by 2 minutes)
- Video contains minor content issues
- Report a little long (less than 3 pages over limit)

Major Issues

Some issues are more substantive on the grade given. Major issues can include:

- File documentation incomplete, incorrect, or missing
- Method documentation incomplete, incorrect, or missing
- Significant grammar issues (use of "I", use of contractions)
- Program output incorrect
- Result insufficiently tested to demonstrate completeness
- Code highly unorganized or tabbed improperly
- Problems with video or audio quality
- Missing requirement
- Report missing important sections
- Video contains major content issues
- Report too long (more than 3 pages over limit)

Self-Evaluation

Students must be able to evaluate their own work, recognizing that requirements are to be completed as written. Part of the report will include a section that describes any minor or major issues with the submission. Students are to indicate any issues with the submission.

Weekly Dropbox

The first draft, final drafts, and videos will all be submitted in a weekly dropbox. Reports that are to be printed must be uploaded by Wednesday of that week; if submitted after that, students must provide a printout to be considered for grading that weekend.

Program Documentation

This section applies only to assignments involving the use of a programming language.

Writing proper documentation is a vital skill for programmers, and yet most computer science courses do not include a grade or requirements for it. As such, this class used the following documentation requirements:

- Every program file must have properly formatted block documentation at the top of the file, as shown in the example below. This is to identify the code, problem, author, credits, and other information.
- Every program function or method must have properly formatted block documentation (indented), to indicate the function's purpose, parameters, and return value. Other information may be provided. The documentation should be consistently indented, and appear directly above the method definition.
- Code must be properly indented, with each new level consistently tabbed.
- Variable names should be descriptive and meaningful. Single-letter variables are acceptable only when the meaning is clear (such as i, x or n). Do not try to abbreviate variable names, as the meaning becomes confusing and ambiguous.
- The border character must form a complete box that is easy to read and separate from the code. The specific character used (*) may be changed.

Program Documentation Example

Include this documentation at the top of every file contained within a project. Items within [brackets] should be replaced with the required content.

```
/*****************************************************************
*                        Name of program                         *
*                                                                *
*   PROGRAMMER:    programmer name                               *
*   COURSE:        course number and course name                 *
*   DATE:          date submitted                                *
*   REQUIREMENT:   Assignment number                             *
*                                                                *
*   DESCRIPTION:                                                 *
*   Include a paragraph describing contents of file              *
*                                                                *
*   COPYRIGHT: This code is copyright (C) 2025 programmer name   *
*   and instructor name.                                         *
*                                                                *
*   CREDITS:                                                     *
*   Include a list of sources used, including Artificial         *
*   Intelligence                                                 *
*                                                                *
*****************************************************************/
```

Function Documentation Example

Include this documentation for every function contained within an object, including the constructor. The placement of the documentation should be directly over the function definition, not the prototype.

```
/*********************************************************
*   FUNCTION:      function name                           *
*   DESCRIPTION:   brief description of function         *
*   PARAMETERS:    list parameters                         *
*   RETURN VALUE:  describe the return value, if any      *
*********************************************************/
```

Writing Reports

Report Resubmission Instructions

1. Read over the instructor comments and the rules below. Ask for clarification in meaning, or if the handwriting is difficult to read.
2. A score of 20 does not mean the paper is perfect in every way. Even 20/20 reports will still have minor edits to fix before recording the video. Complete these edits for the proof, but there is no need to resubmit.
3. If necessary, consult the YCP Writing Center for help in completing the edits.
4. Complete the edits as instructed. Feel free to bring a draft to the instructor for a quick check.
5. Make a printout of the revised changes. Staple the new printout on top of the old printout with comments.
6. Upload the revised file to the report dropbox on Canvas.
7. Give the stapled printouts to the instructor. The printouts are the indication that the report was resubmitted.

The Rules

- Make sure your name is on the report document. No name, no credit.
- Anything written in RED is an edit that needs to be fixed. Comments written in BLUE indicate questions or directions to explore, but not requested edits.
- "Resubmit" on paper means "Allowing for resubmit for full credit."
- Write in the third person format. Do not use "I" or "you." "We" can be acceptable if used correctly. (Exception: "I" may be freely used in the Self Evaluation section, but no other.)
- Do not use contractions in a formal report. Contractions are a figure of *speech*, not writing.
- A paragraph tends to be 3-5 sentences, more if necessary. A paragraph is not just 1 or 2 sentences.
- Use sections to organize content for readability.
- Use consistent formatting throughout document, including font, lists, size, and spacing.
- Keep the written comments for the first draft. It will be stapled to the second draft, so the instructor can see the changes made, and shows it was submitted on time.
- In the revision for the self-evaluation, mention the edits requested by the instructor and what was done to fix the problems (such as visit the writing center).

Instructor's Class Schedule (Professor Zeller)

Dean Zeller's weekly schedule is as follows:

CRN	COURSE	TITLE	CRED	TYPE	DAYS	TIMES	ROOM	INSTRUCTOR	SEATS	OPEN	TOT ENRL	BEGIN-END
15646	CS101.102	Fund Comp Sci I	3.00	LEC	T R	11:00AM-12:15PM	KEC 123	Zeller D	12	0	12	08/25/25-12/16/25
15648	CS201.101	Fund Comp Sci II	3.00	LEC	M W	08:00AM-09:15AM	KEC 119	Zeller D	12	0	13	08/25/25-12/16/25
15649	CS201.102	Fund Comp Sci II	3.00	LEC	M W	01:00PM-02:15PM	KEC 123	Zeller D	12	1	11	08/25/25-12/16/25
15652	CS340.101	Program Lang Desgn	3.00	LEC	T R	08:00AM-09:15AM	KEC 119	Zeller D	20	3	17	08/25/25-12/16/25
15659	CS456.101	Social/Prof Issues	3.00	LEC	T R	09:30AM-10:45AM	KEC 120	Zeller D	16	0	16	08/25/25-12/16/25

Where is Dean Zeller?

	Monday	Tuesday	Wednesday	Thursday	Friday
8:00	CS201 -- Fundamentals II 8:00-9:15 MW in KEC119	CS340 -- Language Design 8:00-9:15 TR in KEC119	CS201 -- Fundamentals II 8:00-9:15 MW in KEC119	CS340 -- Language Design 8:00-9:15 TR in KEC119	
8:15					
8:30					
8:45					
9:00					
9:15					
9:30		CS456 -- Ethics 9:30-10:45 TR in KEC120		CS456 -- Ethics 9:30-10:45 TR in KEC120	
9:45					
10:00	Office Hours for Students 10:00-12:00 in KEC136 (also check classrooms)		Office Hours for Students 10:00-12:00 in KEC136 (also check classrooms)		
10:15					
10:30					
10:45					
11:00		CS101 -- Fundamentals I 11:00-12:15 TR in KEC123		CS101 -- Fundamentals I 11:00-12:15 TR in KEC123	
11:15					
11:30					
11:45					
12:00					
12:15					
12:30					
12:45					
1:00	CS201 -- Fundamentals II 1:00-2:15 MW in KEC119	Office Hours for Students 1:00-2:00 in KEC136 (also check classrooms)	CS201 -- Fundamentals II 1:00-2:15 MW in KEC119	Office Hours for Students 1:00-2:00 in KEC136 (also check classrooms)	
1:15					
1:30					
1:45					
2:00					
2:15					
2:30					
2:45					
3:00					
3:15					
3:30		Block Time reserved for Faculty Meetings 3:30-5:00		Block Time reserved for Faculty Meetings 3:30-5:00	
3:45					
4:00					
4:15					
4:30					
4:45					
5:00					
5:15		Also available for: CS101 -- Fundamentals I 5:00-6:15 TR in KEC130		Also available for: CS101 -- Fundamentals I 5:00-6:15 TR in KEC130	
5:30					
5:45					
6:00					
6:15					
6:30					