UNIVERSITY OF WATERLOO
FACULTY OF ENGINEERING
Department of Electrical & Computer Engineering

Welcome to ECE 150

Fundamentals of programming

Prof. Hiren Patel, Ph.D.
Prof. Werner Dietl, Ph.D.
Douglas Wilhelm Harder, M.Math. LEL

Outline

- In this topic, we will
  - Look at the purpose of this course
  - Go over the main sections of the course syllabus
  - Give you a warning about plagiarism
  - Discuss using generative artificial intelligence in this course

Course outcomes

- By the end of this course, you will be able to
  - Program computers to carry out operational tasks using the C++ language
  - Demonstrate ability to perform both procedural programming and object-oriented programming
  - Develop and implement programs to solve electrical and computer engineering problems
  - Demonstrate ability to test and debug programs

Territorial acknowledgement

We acknowledge that we live and work on the traditional territory of the Attawandaron (Neutral), Anishnaabeg and Haudenosaunee (Iroquois) peoples. The University of Waterloo is situated on the Haldimand Tract, the land promised to the Six Nations of the Haudenosaunee (the Cayuga, Mohawk, Oneida, Onondaga, Seneca and Tuscarora peoples) that includes six miles on each side of the Grand River.

Course topics

- The topics will be broken into six sections:
  1. Programming fundamentals:
     - Syntax, local variables and types, functions and parameters, various operators, control statements (conditional and looping statements) and arrays
  2. Addresses and pointers
  3. Algorithms
  4. Classes
  5. Linked lists
  6. Inheritance and polymorphism

Course websites

- Lecture material and notes are available
  - https://ece.uwaterloo.ca/~ece150/
- Additional material would be available on LEARN
  - https://learn.uwaterloo.ca/

Course instructors

- Prof. Hiren Patel, Ph.D., P.Eng.
  - Email: hiren.patel@uwaterloo.ca
  - Office: **E5** 4018
  - Website: https://caesr.uwaterloo.ca

- Prof. Werner Dietl, Ph.D.
  - Email: wdietl@uwaterloo.ca
  - Office: **EIT** 4007
  - Website: https://ece.uwaterloo.ca/~wdietl/

- Douglas W. Harder, M.Math., LEL
  - Email: dwharder@uwaterloo.ca
  - Office: **E3** 3157
  - Website: https://ece.uwaterloo.ca/~dwharder/

Course grading scheme

- This course has four graded projects and two graded examinations:
  - A mid-term and a final examination
- If your final examination grader is greater than any one of these grades, that higher grade replaces the lower one
- Late policy
  - No late submissions will be accepted
  - Any missed grade is automatically replaced by your final examination grade
    - We do not require a Verification of Illness form or a doctor's note or other documentation
  - There is no make-up mid-term examination

Course grading scheme

- Let your grades be P_1, P_2, P_3, P_4, M and F, each out of 100
  - If F is greater than any other grade,
    the other grade is replaced* by F
  - Your examination grade E is:

$$E = \tfrac{1}{4}M + \tfrac{3}{4}F$$

    - The final is worth three times that of the mid-term
  - Your project grade P is:

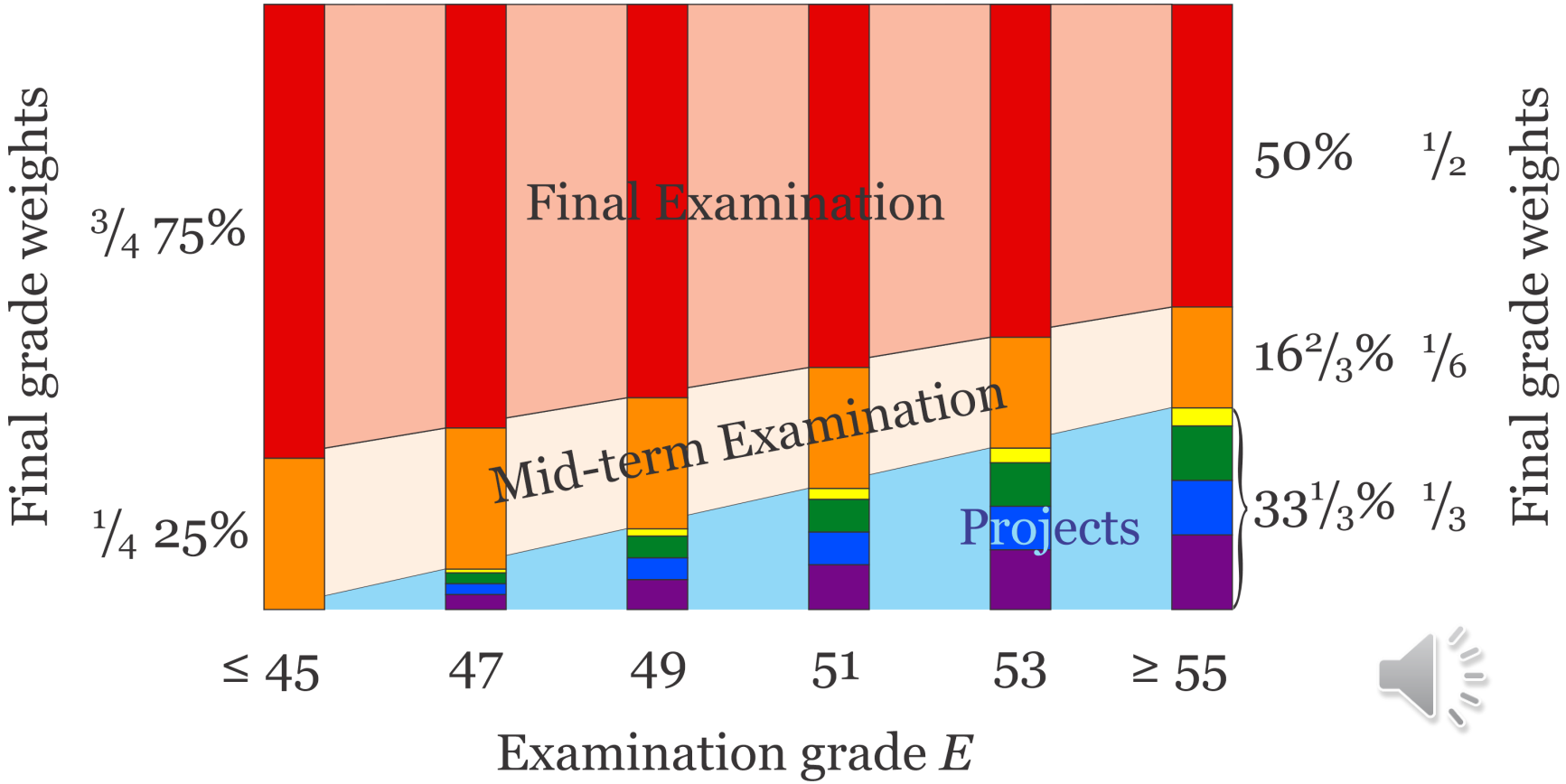
$$P = 0.09P_1 + 0.27P_2 + 0.27P_3 + 0.37P_4$$

- If $E \geq 55$, all your projects count and your final grade is $\tfrac{2}{3}E + \tfrac{1}{3}P$
- If $E \leq 45$, none of your projects count, so your final grade is E
- For $45 < E < 55$, each grade n above 45
  allows your projects to count $10n\%$ of $\tfrac{1}{3}P$ towards your final grade
  - Your final grade will be $E = (\tfrac{1}{30}n)P + (1 - \tfrac{1}{30}n)E$

* Unless the grade is the result of Policy 71

Course grading scheme

- Visually, you can see the examinations and project weights here:



Communication with instructor

- E-mail is a good way to reach us, but please follow a few simple guidelines
  - Only use your `uWaterloo` email address to send email
    - No forwarding through `gmail` or others
  - Put "`ECE150`" in the subject to increase the priority
  - Be concise and clear when describing your concern
  - Be patient, we will do our best to respond to you quickly
    - You can always contact the WEEF teaching assistants (TAs)

Collaboration versus plagiarism

- We encourage students to work together
  - Teaching others is one of the best way to ensure you have a mastery of the subject matter
  - You may assist your friends
  - You should only examine another student's code if you believe you have a reasonable solution, and they are clearly having issues that you have already solved
  - You should help the student you are helping understand the problem
    - Never give the student the correct solution—this will not help that individual and they will become dependent on you

Academic misconduct

- It is considered to be *academic misconduct* if:
  - You send your solutions in any format (including a verbal reading thereof) to anyone else, even if they then forward those solutions to a third party
  - You submit code that you were not the sole author thereof
  - You edit or dictate someone else's code
  - You search the web for a similar problem with a posted solution
  - You get a solution from a student in a previous year and submit it, or something very similar with only cosmetic or minor changes
  - **You post your code on GitHub or replit.com or any other publicly accessible web site and someone else downloads and submits your code**
  - Leave your computer unattended and a peer accesses that computer to access your code

Academic misconduct

- This is critical:

DO NOT SEND OR COMMUNICATE YOUR SOLUTIONS TO YOUR PEERS

- In every instance of plagiarism in the past two years, there was clear evidence that code was shared or copied from a common source

Academic misconduct

- A few points to remember:
  - You will likely think there is only one solution,

    so sending your code will help your friend fix their code

    Problem: There are many solutions, and your solution probably will not help your peer fix that person's own approach
  - Your friend promised not to plagiarize

    Problem: When a student is under stress, that student will forget such promises, and as the deadline approaches, the pressure will build to the point of being unbearable

Academic misconduct

- A few points to remember:
  - This may be your best friend from secondary school, or even elementary school, or it may be a friendship you've recently cultivated

    Problem: We have had cases where students who knew each other since Grade 1 ended up in a case of plagiarism
  - Your friend plagiarized your code, got caught, and then swears up and down and left and right that no code was copied

    Problem: You have just lost at least 10% of your final grade, and you will be prone to believing your friend
    - This will ruin your first-year experience
    - Remember, each plagiarism case was sent to the Associate Dean of Undergraduate studies, and that individual concurred that plagiarism was evident
    - Also, the evidence for each plagiarism case was shown to a 2A class representative, and that student concurred that plagiarism was evident

Academic misconduct

- A few points to remember:
  - You may think: but my code was perfect, why am I being penalized

    Issue: You have helped someone else plagiarize, so you are just as much to blame as the other person. Also, you are getting something out of the deal:
    - You feel good because that student immediately shows you gratitude
    - You've just scratched that student's back,

      they will have the opportunity in the future to scratch yours
  - You may be from a culture that pressures you to help your peers

    Solution: So help them, but do not send them your code

    It requires much more effort to help them, but that effort will pay off

Academic misconduct

- Important:
  - Help your friends, and if necessary, you may have to lose some sleep, but do not send them your code, even after the deadline
  - Helping your friends will also help you
  - In many cases last year, it was clear the intention of the student supplying the solution was trying to help the other student

- This is critical:

DO NOT SEND OR COMMUNICATE YOUR SOLUTIONS TO YOUR PEERS

Penalties

- If you are found guilty of academic misconduct, the result is
  - Zero on the entire assessment,
    and that zero will not be replaced by the final examination grade
  - A penalty of 5% from your final grade per infraction
- On subsequent offences, the penalties may increase:
  - A required selection of courses in ethics
  - A failing grade in the course
  - A two-year suspension
  - A 7-year suspension
  - Expulsion

Example

- Suppose you copied your friend's code on Project 3:
  - Suppose you got:
    - 43% on the mid-term examination
    - 54% on the final examination
    - 86% on the projects, including Project 3
  - Suppose your friend got:
    - 75% on the mid-term examination
    - 82% on the final examination
    - 95% on the projects
- If you hand in the plagiarized Project 3, your grade is 61
- If you don't hand in the project, your grade is 59
  - The final examination replaces the 0 in Project 3
- If you are caught, with penalties, your grade is now 51

Fairness

- To be fair to all students, all accusations of plagiarism will have their names redacted and shown to the class representatives
  - The class representatives will have the option of suggesting that a particular case is not a clear demonstration of plagiarism

- Last year, we showed all cases to the class representatives, and they disputed none of the accusations
  - The class representatives from your class can request to see those cases, as well
  - Last year, all accusations resulted in a finding of guilty

- We do not malevolently try to go after every possible case, we only file accusations when it is clear plagiarism occurred
  - We need you to understand that there are consequences…

How to succeed

- Watch the lectures and take notes
  - They should make it easier to understand the main concepts
- Practice programming
  - There is no substitute for solving problems using programming
- Clarify confusions *early*
  - Seek help from your peers, WEEF tutors and TAs
  - Ask instructor
- Don't just view the information once
  - Repeatedly reviewing the content will help you remember concepts
- Work ahead
  - A large portion of the course's lecture material is already available for you to study in advance

Use of artificial intelligence

- There are amazing:
  - ChatGPT, Github co-pilot and replit.com ghostwriter
- Many of the problems given in your undergraduate studies can be solved by one of these tools, but this doesn't help you learn
- You may not use any artificial intelligence tools for the projects
- You can use these tools to learn the C++ syntax, and to find issues with your solutions to various problems

Use of artificial intelligence

- For example, using ChatGPT (because it the most accessable):
  - Ask it "Why does this code not compile?" and then cut-and-paste your code into ChatGPT
    - Usually it will find your syntax errors
    - It may also make some other suggestions
  - If your code does not seem to work in solving a problem, ask it "Why does this not solve ...." or "Why does this not find ..."
  - You can even ask: "Do you have any suggestions to improve this code?"

Use of artificial intelligence

- Remember: engineering is about problem solving, but artificial intelligence usually does solve new and unique problems
  - You must develop your problem-solving toolkit starting with problems for which we know there is a solution
  - If you cheat yourself by using artificial intelligence instead of your own intelligence, you will not be able to tackle real-world unique and novel problems
    - You will be an artificial intelligence technician, not an engineer

Summary

- Following this lesson, you now
  - Understand this course's outcomes and topics
  - Know the grading scheme
  - Understand the policies related to plagiarism, penalties, and late submissions
  - Have been provided suggestions on how to succeed in this course
  - Understand how to use artificial intelligence to help you succeed

References

[1] https://ugradcalendar.uwaterloo.ca/courses/ECE/150
[2] https://ece.uwaterloo.ca/~ece150/
[3] https://openai.com/ for ChatGPT
[4] https://en.wikipedia.org/wiki/GitHub_Copilot
[5] https://en.wikipedia.org/wiki/Generative_artificial_intelligence

Colophon

These slides were prepared using the Georgia typeface. Mathematical equations use Times New Roman, and source code is presented using `Consolas`.

The photographs of lilacs in bloom appearing on the title slide and accenting the top of each other slide were taken at the Royal Botanical Gardens on May 27, 2018 by Douglas Wilhelm Harder. Please see

https://www.rbg.ca/

for more information.

Disclaimer

These slides are provided for the ECE 150 *Fundamentals of Programming* course taught at the University of Waterloo. The material in it reflects the authors' best judgment in light of the information available to them at the time of preparation. Any reliance on these course slides by any party for any other purpose are the responsibility of such parties. The authors accept no responsibility for damages, if any, suffered by any party as a result of decisions made or actions based on these course slides for any other purpose than that for which it was intended.