ENSF 692: Programming Fundamentals for Data Engineers

Welcome to ENSF 692: Programming Fundamentals for Data Engineers. This course will equip you with essential Python programming skills needed for data engineering applications. I will guide you through this journey of developing coding proficiency for data manipulation, analysis, and problem-solving.

S **by Sarah Shah**

Why Learn Python?

Data Science Powerhouse

Python has become the dominant language in data science and engineering, with robust libraries and frameworks specifically designed for data manipulation and analysis.

Teaching-Oriented Design

Originally developed as a teaching language, Python emphasizes readability and simplicity, making it ideal for beginners while remaining powerful for experts.

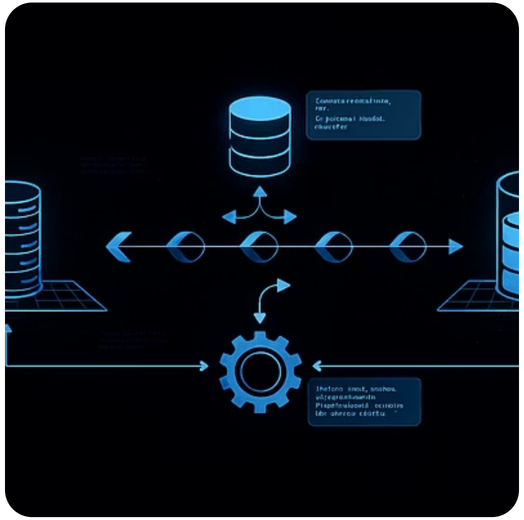
"Executable Pseudocode"

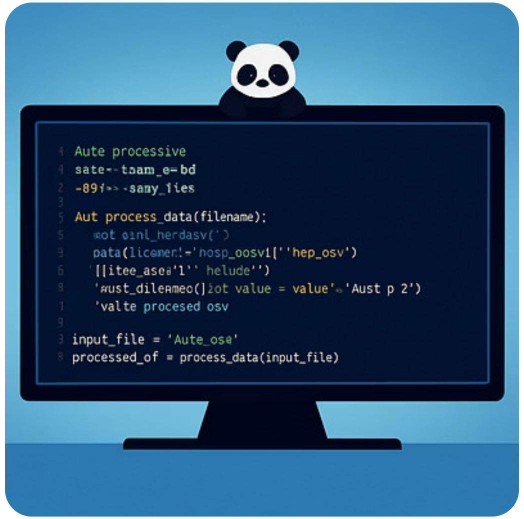
Python's syntax is so clear and intuitive that it's often called "executable pseudocode," allowing you to focus on problem-solving rather than language complexities.

Interpreted Language

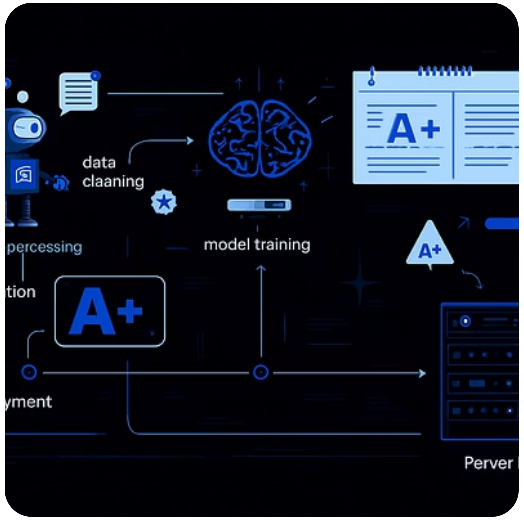
As an interpreted language, Python offers immediate feedback during development, facilitating an iterative learning process ideal for building programming skills.

Python in Data Engineering





```
Aute processive
sate-- taam_e=bd
-89i>> -samy_1ies

Aut process_data(filename):
   wot oinl_herdasv(')
   pata(licener!='hosp_oosv1[''hep_osv')
  '[|itee_asea'1'' helude'')
   'wust_dileAmeo(]iot value = value'-'Aust p 2')
   'valte procesed osv

input_file = 'Aute_ose'
processed_of = process_data(input_file)
```



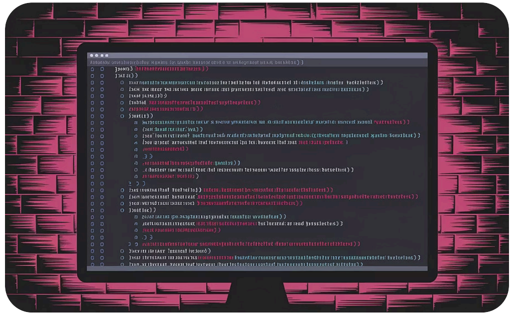
Python has become the lingua franca of data engineering due to its versatile ecosystem of libraries and frameworks. From data extraction and transformation to analysis and visualization, Python provides tools that streamline the entire data pipeline process. In this course, you'll learn how to leverage these capabilities to solve real-world data engineering challenges.

Libraries like Pandas, NumPy, and Matplotlib form the foundation of Python's data ecosystem, while frameworks like Apache Airflow enable complex workflow orchestration. Understanding these tools will be essential to your success as a data engineer.

Course Objectives

Problem-Solving Skills

Analyze and solve logical problems through coding, developing critical thinking abilities essential for data engineering.

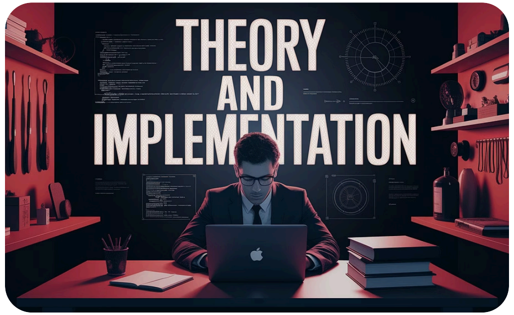


Python Fundamentals

Develop and implement programs using the foundational structures of Python, building a strong programming foundation.

Data Applications

Apply coding techniques for data input, manipulation, analysis, and output, preparing you for real-world data challenges.



Engineering Solutions

Translate engineering requirements into coded solutions, bridging the gap between theory and practical implementation.

Meet Your Teaching Team

Instructor

Sarah Shah will lead course instruction and oversee all aspects of the learning experience. Contact via email at sarah.shah1@ucalgary.ca for course-related inquiries.

Teaching Assistant

Amirreza Hosseini will provide additional support during labs and assignment grading. Reach out at amirreza.hosseini@ucalgary.ca for assistance with exercises and projects.

Course Format

Lectures

Instruction will primarily be conducted through **interactive textbook notebooks** rather than traditional slides. This hands-on approach allows for immediate application of concepts.

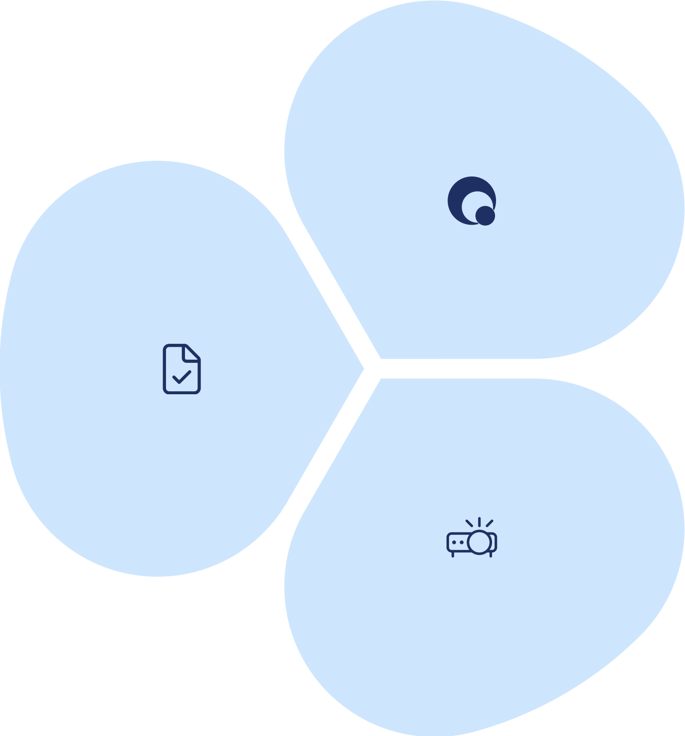
You'll follow along with coding examples and participate in live demonstrations of Python techniques relevant to data engineering challenges.

Labs

Lab sessions provide structured time to complete Python exercises under guidance from the teaching team. These **practical sessions reinforce lecture concepts**.

Labs also offer dedicated time to work on assignments and project components with immediate feedback available from the TA.

Assessment Breakdown



Assignments

50% of final grade

Five assignments with varying weights

Quizzes

20% of final grade

Two in-person quizzes, D2L quizzes

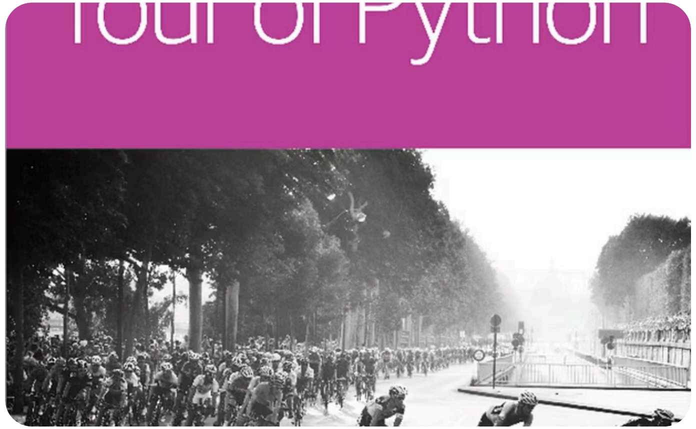
Project

30% of final grade

Must pass to pass the course

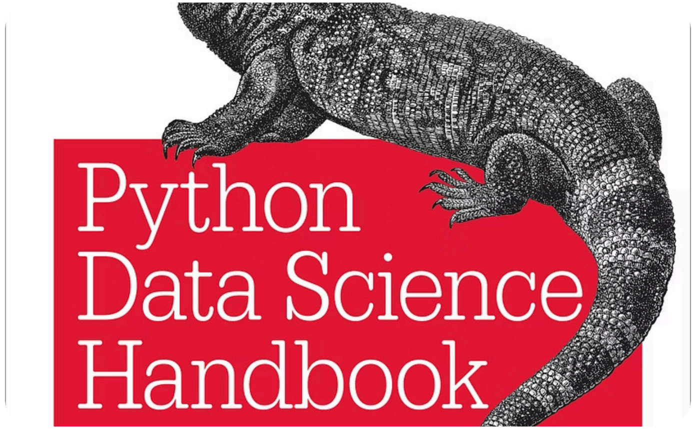
Note that there are no midterm or final exams in this course. Accommodations are available through **Student Accessibility Services**. Letter grade conversion details can be found in the course outline.

Course Textbooks



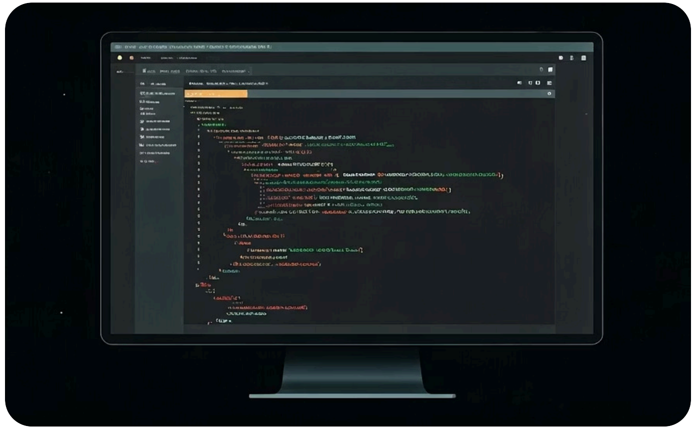
Whirlwind Tour of Python

Also by Jake VanderPlas, this concise guide provides a rapid introduction to Python programming fundamentals and syntax.



Python Data Science Handbook

By Jake VanderPlas, this comprehensive resource covers essential Python libraries and techniques for data analysis, visualization, and machine learning.



Interactive Access

Access these materials through D2L as interactive notebooks using Google Colab, Anaconda's Jupyter Lab/Notebook, or VSCode.

Communication Methods

D2L Platform

Primary communication channel for course announcements, materials, and updates. Check regularly for important information and new resources.

Email Communication

Use your ucalgary.ca email for correspondence. Include your full name, ID number, and course. Expect responses within 48 hours on business days.

Discussion Boards

Two dedicated boards: **Content Questions** for course material inquiries and **Community Building** for sharing interesting topics and interacting with your peers. You are encouraged to answer each other's questions.

All communication channels are moderated by the instructor and the TA. Respect and professional conduct are expected in all interactions, reflecting the standards of the engineering profession.

Academic Integrity

Professional Ethics

Engineers adhere to a code of ethics and professionalism

Appropriate Collaboration

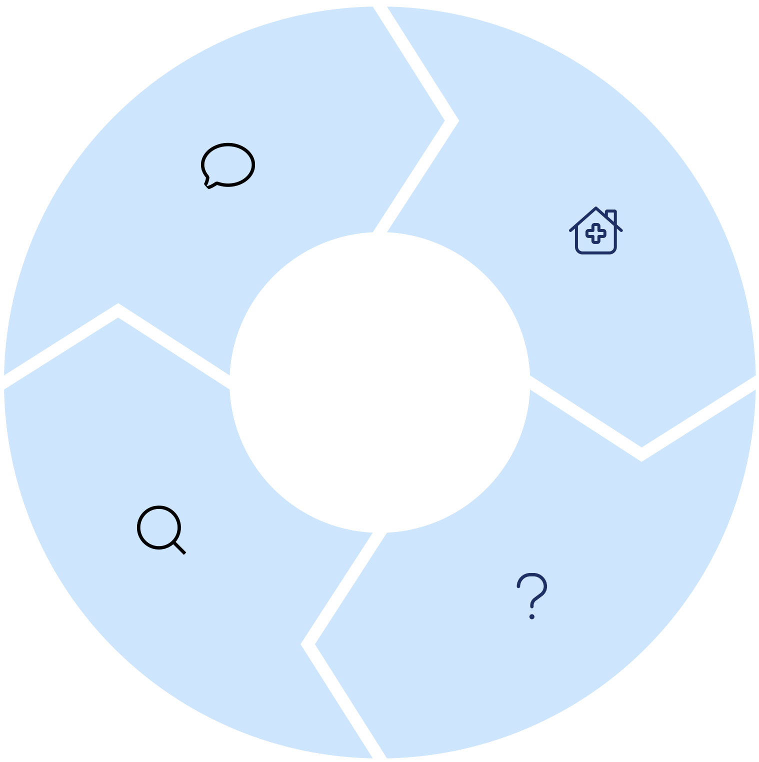
Understand the difference between peer assistance and misconduct

Intellectual Property

Respect that code is copyrighted and protected

As future engineers, you're expected to maintain high standards of integrity in your academic work. This includes using AI tools appropriately, citing sources correctly, and completing assessments independently unless otherwise specified. Additional resources on academic integrity are available on the university website.

Ensuring Success



Take Ownership

Take responsibility for your own learning journey and progress

Regular Practice

Consistently work with Python to build skills and confidence

Use Resources

Take advantage of all available learning materials and support

Ask Questions

Reach out when struggling with concepts or assignments

Success in this course requires active engagement with the material and a proactive approach to learning. Programming skills develop through consistent practice and problem-solving. Additional resources to support your learning journey are posted on D2L.

Today's Exercises

Icebreaker Activities

Take the initiative to get to know your colleagues. You will partner up with them during the course. Software development and data engineering are team sports, so building connections now will enhance your learning experience throughout the course.

Discussion Board Introduction

Introduce yourself on the designated discussion board and respond to at least one other post. Share your background, interests, and what you hope to gain from this course.

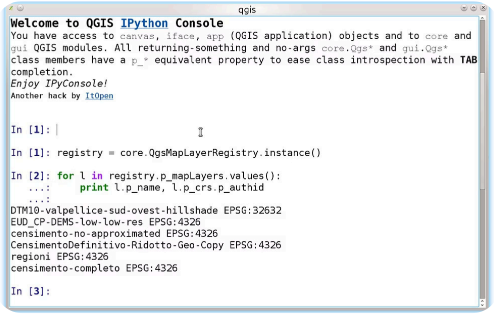
Python Environment Overview

Explore the Python environments and textbook resources that will be used throughout the course. Get familiar with the tools and activities that will support your learning journey.

Running Python

Python Interpreter

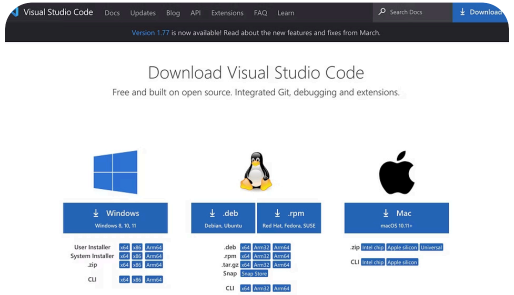
The standard Python interpreter provides a direct way to execute Python code through a command-line interface, ideal for quick scripts and testing.



```
Welcome to QGIS IPython Console
You have access to canvas, iface, app (QGIS application) objects and to core and
gui QGIS modules. All returning-something and no-args core.Qgs* and gui.Qgs*
class members have a p_* equivalent property to ease class introspection with TAB
completion.
Enjoy IPyConsole!
Another hack by ItOpen


In [1]: 

In [1]: registry = core.QgsMapLayerRegistry.instance()

In [2]: for l in registry.p_mapLayers.values():
   ...:     print l.p_name, l.p_crs.p_authid
   ...:
DTM10-valpellice-sud-ovest-hillshade EPSG:32632
EUD_CP-DEMS-low-low-res EPSG:4326
censimento-no-approximated EPSG:4326
CensimentoDefinitivo-Ridotto-Geo-Copy EPSG:4326
regioni EPSG:4326
censimento-completo EPSG:4326

In [3]:
```

IPython

An enhanced interactive Python shell offering making it excellent for exploratory programming.



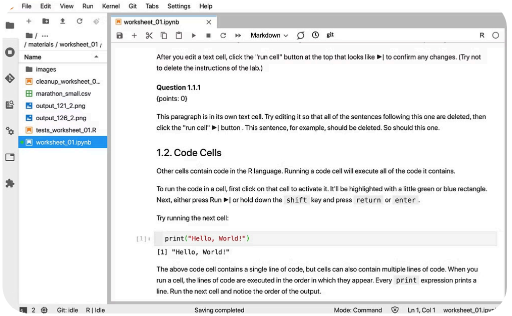
Version 1.77 is now available! Read about the new features and fixes from March.

Download Visual Studio Code

Free and built on open source. Integrated Git, debugging and extensions.

VSCode

A powerful code editor with extensive Python support, including debugging, linting, and version control integration, preferred for larger projects.



After you edit a text cell, click the "run cell" button at the top that looks like ▶| to confirm any changes. (Try not to delete the instructions of the lab.)

Question 1.1.1
{points: 0}

This paragraph is in its own text cell. Try editing it so that all of the sentences following this one are deleted, then click the "run cell" ▶| button . This sentence, for example, should be deleted. So should this one.

1.2. Code Cells

Other cells contain code in the R language. Running a code cell will execute all of the code it contains.

To run the code in a cell, first click on that cell to activate it. It'll be highlighted with a little green or blue rectangle. Next, either press Run ▶| or hold down the `shift` key and press `return` or `enter`.

Try running the next cell:

```
[1]: print("Hello, World!")
[1] "Hello, World!"
```

The above code cell contains a single line of code, but cells can also contain multiple lines of code. When you run a cell, the lines of code are executed in the order in which they appear. Every `print` expression prints a line. Run the next cell and notice the order of the output.

Python Notebooks

Interactive documents combining code, visualizations, and narrative text, perfect for data analysis workflows and educational content.

Python Development Environments

Integrated Development Environments (IDEs)

Comprehensive tools like PyCharm and VSCode provide robust features for Python development, including code completion, debugging, and project management. These environments are ideal for larger projects and professional development.

IDEs offer integrated terminal access, version control, and extension ecosystems that enhance productivity. They typically include linting tools that help identify potential issues in your code before execution.

Notebook Environments

Jupyter Notebooks and Google Colab provide interactive computing environments that combine code execution with rich text, visualizations, and media. These are particularly well-suited for data analysis and educational purposes.

Notebooks allow for incremental development and documentation, with the ability to execute code cells individually and see results immediately. This makes them excellent for experimentation and learning.

Upcoming Activities

Installation

Complete installations for running Python, including Python interpreter, VSCode, and JupyterLab

Basic Syntax

Learn fundamental Python syntax and semantics to build a strong foundation

Assignment 1

Begin work on your first programming assignment to apply new skills

These upcoming activities will help you establish the technical environment and foundational knowledge needed for success in this course. Ensuring your development environment is properly configured is crucial for completing assignments and following along with course materials.

Programming Best Practices

Principle	Description	Python Example
Readability	Write clear, self-documenting code	Use descriptive variable names and follow PEP 8 style guide
Modularity	Break code into reusable functions	Define functions for specific tasks rather than monolithic scripts
Documentation	Comment code appropriately	Use docstrings and inline comments to explain complex logic
Testing	Verify code works as expected	Implement unit tests using pytest or unittest frameworks
Version Control	Track changes systematically	Use Git for managing code versions and collaboration

Your Path to Python Proficiency

12

Weekly Labs

Hands-on practice sessions to reinforce concepts and build skills under guidance

5

Assignments

Progressive challenges that build your programming capabilities step by step

2

Quizzes

Knowledge checks to ensure understanding of core Python concepts

1

Final Project

Comprehensive application of all skills in a real-world data engineering scenario

Assignment Details

Preparation

Create Python scripts and notebooks to accomplish predefined tasks that demonstrate your understanding of course concepts.

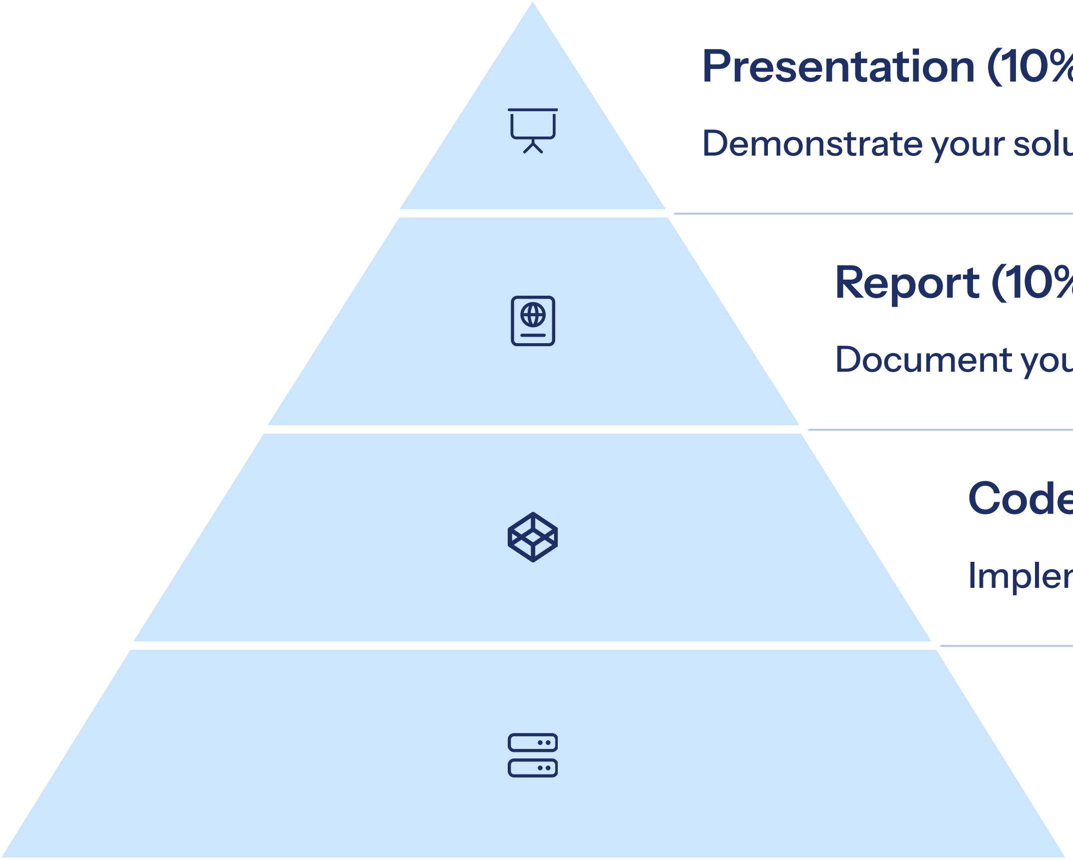
Submission

Submit the HTTPS link to your repository or notebook via the designated D2L dropboxes by the specified deadlines.

Evaluation

Assignments are assessed based on functionality, code quality, documentation, and adherence to requirements.

Project Components



Presentation (10%)

Demonstrate your solution

Report (10%)

Document your approach

Code & UI (60%)

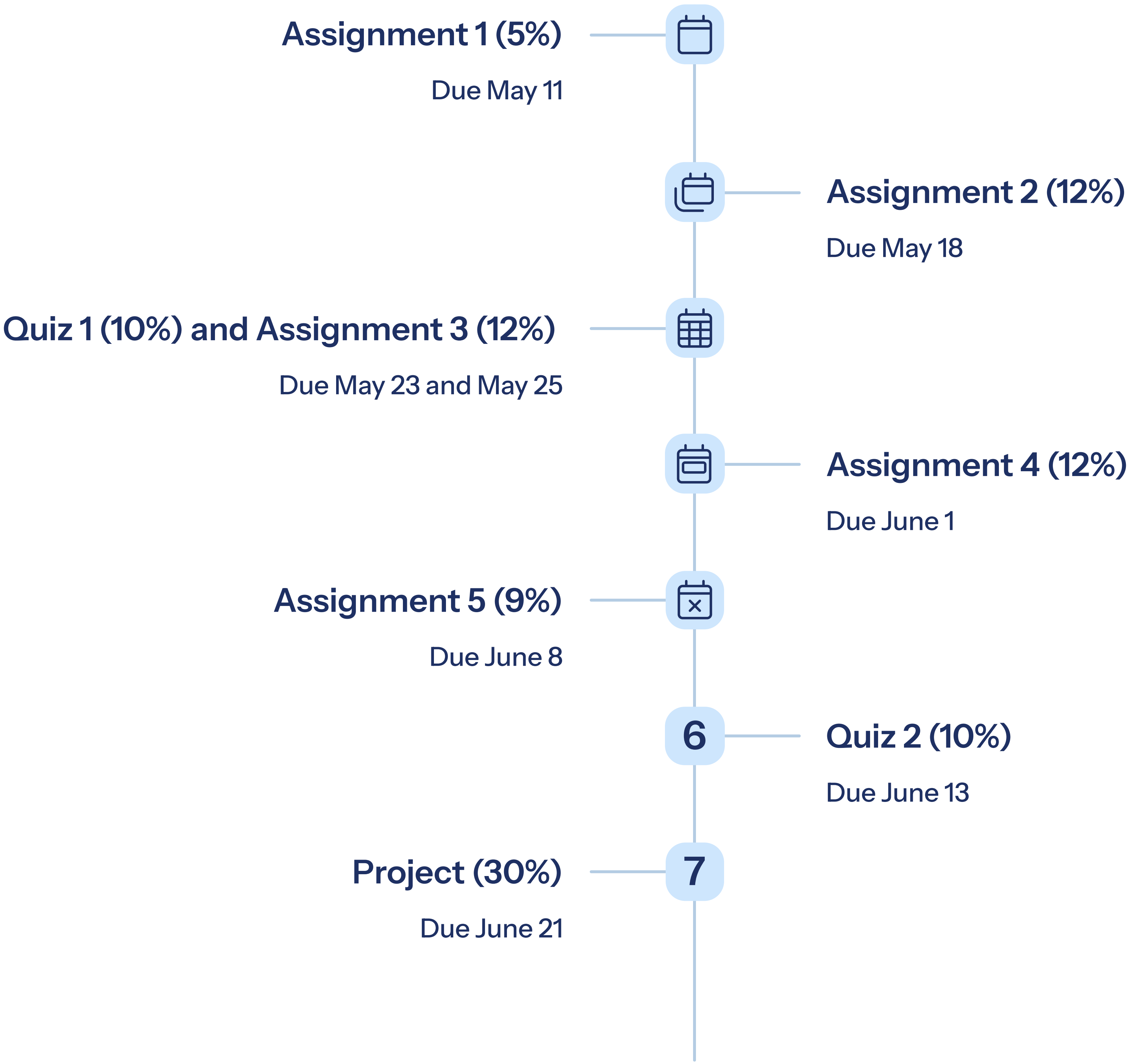
Implementation, interface, execution

Data Handling (20%)

Foundation of your solution

The course project is a comprehensive assessment that integrates all the skills developed throughout the course. You'll need to handle data effectively, implement code solutions, create a user interface, document your work thoroughly, and present your findings.

Deliverable Schedule



Assignment 1 (5%)

Due May 11

Assignment 2 (12%)

Due May 18

Quiz 1 (10%) and Assignment 3 (12%)

Due May 23 and May 25

Assignment 4 (12%)

Due June 1

Assignment 5 (9%)

Due June 8

6

Quiz 2 (10%)

Due June 13

7

Project (30%)

Due June 21