# LIFE733: Python Programming – Weeks 1–5 Teaching and Assessment Plan

# Integrating Responsible GPT Use in LIFE733

## Philosophy: GPT-Aware, Not GPT-Dependent

This course acknowledges that students may use tools like ChatGPT and other generative AI. Rather than banning them, LIFE733 teaches students how to use these tools responsibly. The goal is to build AI literacy, not AI dependence.

## Course-Wide Guidance (Canvas Resource)

A dedicated Canvas page will be created titled:

"How to use GPT for LIFE733 (without letting it write your code for you)"

This page will outline:

Appropriate Use:  
- Asking GPT to explain syntax or error messages  
- Suggesting test cases or alternative solutions  
- Refactoring or improving readability of code

Inappropriate Use:  
- Submitting GPT-generated code without understanding it  
- Blindly copy-pasting responses  
- Relying on GPT to solve assessment tasks

How to Check GPT Output:  
- Run it on edge cases or unexpected inputs  
- Deliberately break and test the output  
- Ask GPT to explain its own code, then re-express it in your own words

## Week-by-Week Integration

Week 1: Introduction to GPT in programming: what it is, how it helps, when it fails. Live demo of GPT-generated code for a basic task.

Week 2: Debugging triangle classifiers: Ask GPT for help, but verify its fix. Reflection: “What did GPT suggest? Was it right?”

Week 3: Binning task: Ask GPT to classify scores into categories. Compare with manual solution. Discuss GPT errors.

Week 4: Prompt GPT to write a DNA translation function. Test it with invalid inputs. Reflection: "Where did it fail?"

Week 5: Regex hallucination: show an incorrect GPT-generated motif matcher. Ask students to fix and explain why it was wrong.

## Assessment Integration

Assessment 1:  
- Students must include a declaration in each `.py` file, e.g.:  
 I used GPT to help with [X]. I verified the output by [Y]. One thing GPT got wrong was: [Z].

- At least one task will involve buggy code students are allowed to debug with GPT, but they must explain why their fix is correct.

Assessment 2:  
- For regex motif matching, students must:  
 - Explain how they constructed their regex  
 - Describe how they tested it  
 - Reflect on where GPT might hallucinate or fail

Sample reflection prompt:  
What would GPT get wrong about this task, and how did you avoid it?

## Embedding Best Practice Across the Course

Teaching:  
- In-session demos, GPT debugging walkthroughs, discussion of AI hallucinations

Formative:  
- Tasks where GPT is encouraged, but students must test and reflect

Assessment:  
- Mandatory use declarations, reflection prompts, and buggy code challenges

Canvas:  
- Permanent guidance page, referenced in each week’s module