Programming II (KI/EAPR2)

Beránek Pavel (pavel.beranek@ujep.cz)

1. Course Syllabus

Lesson	Date	Topic
1		Implementation of classes (data fields, methods, constructors)
		Abstract data types (ADT), implementation of sequential data structures (queue, stack, sorted list)
2		Implementation of algorithms over sequential collections - search algorithms
		Implementation algorithms over sequential collections - sorting algorithms
3		Linked structures and their object implementation (linked lists, trees, etc.)
		Implementation of algorithms over linked structures
4		Interfaces (protocols or abstract base classes in Python parlance), iterators and their implementations, interfaces of basic collections
		Inheritance (advantages and disadvantages)

2. Study materials

Github: https://github.com/pavelberanek91/EAPR2/tree/main

Lesson 1

- Object-orienting programming: https://www.freecodecamp.org/news/crash-course-object-oriented-programming-in-python/
- Sequential data structures: https://www.javatpoint.com/ds-stack-vs-queue

Lesson 2

- Search algorithms: https://www.freecodecamp.org/news/search-algorithms-linear-and-binary-search-explained/
- Sorting algorithms: https://www.programiz.com/dsa/sorting-algorithm

Lesson 3

- Linked data structures: https://www.javatpoint.com/tree-vs-graph-data-structure
- Search algorithms on linked structures: https://memgraph.com/blog/graph-search-algorithms-developers-guide

Lesson 4

- Iterators: https://realpython.com/python-iterators-iterables/
- Inheritance: https://realpython.com/inheritance-composition-python/
- Interfaces (Protocols): https://xebia.com/blog/protocols-in-python-why-you-need-them/

3. Requirements

To successfully obtain credit, the student must submit solutions for the following 4 tasks in Python and must be able to discuss them.

Topic 1: Discriminator for Bracket-Balanced Expressions
Implement your own stack to develop a mechanism that determines whether expressions are balanced in terms of brackets. This involves creating a custom stack structure to evaluate expressions for balanced or unbalanced brackets.

Materials: https://www.prepbytes.com/blog/stacks/check-for-balanced-parentheses-in-an-expression/

Topic 2: Comparison of Sorting Algorithms
Develop a Python script that incorporates various sorting algorithms, such as bubble sort, merge sort, and quicksort. Design experiments to evaluate the performance (time complexity) of these algorithms under various scenarios, like differing array sizes and degrees of sortedness. Utilize graphical plots to vividly compare and highlight the efficiency of each algorithm.

Materials: https://www.toptal.com/developers/sorting-algorithms

Topic 3: The Traveling Salesman Problem
Leverage your custom implementation of a graph data structure to model a country with towns connected by distances. Aim to find the shortest route starting from a chosen town to all other towns, minimizing the total distance traveled.

Materials: https://trackobit.com/blog/what-is-a-traveling-salesman-problem-explained

Topic 4: Design a Simple Video Game
Create a video game of any genre to demonstrate effective use of inheritance and class aggregation/composition. The game can either have a command-line interface or a full graphical user interface using a framework like PyGame.

Materials: https://realpython.com/pygame-a-primer/