Aligned subject learning outcomes	· **SLO1 Evaluate the efficiency and effectiveness of data structures and algorithms** · **SLO2 Demonstrate reasoning about efficiency of algorithms** · **SLO4 Design and implement data structures and algorithms**
Group or individual	*Individual*
Weighting	*20%*
Due date	Week 7, Friday 5pm

A **sparse array** is an array in which most elements are empty (in Python, None). A list L can be used to implement such an array efficiently. In particular, for each nonempty cell A[i], we can store an entry (i,e) in L, where e is the element stored at A[i]. This approach allows us to represent A using O(m) storage, where m is the number of nonempty entries in A.

For this assignment, you are to implement a **SparseArray** class based on a linked list. Your class should support the following methods:

- __init__(n) to construct an initially empty SparseArray of size n
- fill(seq) to fill the SparseArray with data from a given sequence seq
- __getitem__(j) and __setitem__(j, e) to provide standard indexing operations
- __len__() to enable len(A)

You must provide:

1. Pseudocode for the methods of the SparseArray class.
2. An analysis of the efficiency of your algorithms for __getitem__(j), __setitem__(j, e), and fill(seq) in terms of **m**, the number of actual non-empty elements, and **n**, the overall size of the array. You must provide justification for your analysis.
3. Python code for the SparseArray class. You may use any underlying linked list implementation of your choice, but should justify whatever you are using.
4. Python code for efficiency experiments, as outlined below.

Experiments

Consider values of **n** that are powers of 10, say 10^3 to 10^{12}. Use different values of **m** to look at different levels of sparseness for each of these values of **n**. In each of these cases, consider the time taken to set and get items in random locations within the SparseArray. Provide the output of these experiments. Examine the running times, and justify at what values of **n** and **m** SparseArray makes more sense than using an actual Python list or dynamic array.

Deliverables

1. **A Word document containing your pseudocode, algorithm analysis, experimental results, and discussion of your findings.**
2. **A ZIP file containing SparseArray implementation in Python and code for running experiments (also include any files that you may need to import).**

Rubric

Criteria	Excellent	Good	Marginal	Unacceptable
Design of SparseArray	**20** Pseudocode for each of the methods is clear and correct.	**15** Pseudocode for most of the methods is clear and correct, with some minor lapses.	**10** Pseudocode for some of the methods is generally correct.	**0** Pseudocode not provided, or incomprehensible and incorrect.
Analysis of algorithms	**20** Correct algorithm analysis of all methods in terms of *n* and *m*, with good justifications.	**15** Correct algorithm analysis of most methods in terms of *n* and *m*, with good justifications.	**10** Reasonable algorithm analysis, with some justifications.	**0** No analysis provided, or failure to justify the analysis.
Implementation of SparseArray	**30** Correct Python implementation of all SparseArray methods using an appropriate linked list structure.	**20** Correct Python implementation for most of the SparseArray methods using a generally appropriate linked list structure.	**15** Some of the SparseArray methods are correctly implemented using a linked list structure.	**0** SparseArray methods are mostly missing, incorrect, or do not use a linked list structure.
Experiments	**20** Appropriate Python code for performing experiments, with good range of different parameters (*n* and *m*). Experimental results are provided in a clear format.	**15** Appropriate Python code for performing experiments, with range of different parameters (*n* and *m*). Experimental results are provided in a clear format.	**10** Some Python code for performing experiments, with different parameters (*n* and *m*). Experimental results are provided.	**0** Incomplete or incorrect Python code for experiments. Fails to provide results.
Discussion	**10** Clear and justified discussion of when the SparseArray implementation would be effective, in terms of the experimental results, and analysis of algorithms.	**7.5** Good discussion of when the SparseArray implementation would be effective.	**5** Some discussion of results.	**0** No discussion of results, or nonsensical discussion.