Completing an Array-API compliant Ragged library

Applicant: Oleksii Hrechykha
Mentor: Ianna Osborne

Python is the most widely used language in high energy physics. However, analysing experimental data often proves problematic, as the data is often incomplete – meaning that not every particle has all the possible parameters for a given experiment. This, in turn, leads to said data being written as a jagged (or ragged) array, where different lines have non-consistent lengths.

When working with jagged data, Awkward array library is often used. But using Awkward creates a new problem, as the library is too general and allows to work with arrays of arbitrary data types, which is not necessary for many HEP purposes. Moreover, this general nature of Awkward arrays prevents the library from being compatible with other array libraries. It requires specialized knowledge and re-learning, making it inconvenient for the general user.

Therefore, in order to make certain parts of Awkward functionality widely accessible, a wrapper library must be written that should be compatible with the Array API. Implementing specific functions for this Ragged array library is the purpose of the project.

Project goals
Implementing the functions currently missing from ragged array library with complete set of tests for each function.

Project deliverables
Complete set of functions currently marked as non-implemented.

Project timeline
Week 1: Recalling things done last year, getting up to speed with the progress made since.
Weeks 2-4: Writing tests and implementation of array objects and creation functions:
- A function that returns transpose of a matrix or a stack of matrices ("TODO 2")
- A function that returns the number of elements in an array("TODO 3")
- A function that computes the matrix product ("TODO 22")
- A function that returns coordinate matrices from coordinate vectors.("TODO 43")
- A function that returns the lower triangular part of a matrix or a stack of matrices ("TODO 46")
- A function that returns the upper triangular part of a matrix or a stack of matrices ("TODO 47").

Weeks 5-7: Writing tests and implementation of linear algebra functions:
- A function that computes matrix product ("TODO 110")
- A function that transposes a matrix or a stack of matrices ("TODO 111")
- A function that returns tensor contraction of two matrices over specific axes ("TODO 112")
- A function that computes the vector dot product of two arrays ("TODO 113").

Weeks 7-9: Writing tests and implementation of array manipulation functions:
- A function which broadcasts one or more arrays to a specified shape ("TODO 115")

- A function which reverses the order of elements in an array along the given axis while preserving the shape of the array ("TODO 118")
- A function which permutes the axes of an array ("TODO 119")
- A function which reshapes an array without changing its data (“TODO 120”)
- A function which rolls array elements along a specified axis while following API guidelines for elements which roll beyond first or last position (“TODO 121”)
- A function which joins a sequence of arrays along a new axis (“TODO 123”).

Week 10: Reporting on the progress made.
Weeks 1-10: Addressing miscellaneous issues that are likely to arise over the course of the project

References:

Ragged array library https://github.com/scikit-hep/ragged