# Assignment 2

CS329e - Elements of Software Design
Collapsing Intervals
(100 points)
Due Date on Canvas and Gradescope

## 1 Description

An interval on the number line is denoted by a pair of values like so: (3, 8). Our intervals are closed. So this interval represents all numbers between 3 and 8 inclusive. The first number is always going to be strictly less than the second number. Normally in mathematics we represent a closed interval with square brackets. But in our program we will represent an interval by means of a tuple and tuples in Python are represented with parentheses.

If we have two intervals like (7, 12) and (4, 9), they overlap. We can collapse overlapping intervals into a single interval (4, 12). But the following two intervals (-10, -2) and (1, 5) are non-intersecting intervals and cannot be collapsed. The aim in this assignment is take a set of intervals and collapse all the overlapping intervals and print the smallest set of non-intersecting intervals in ascending order of the lower end of the interval and then print the intervals in increasing order of the size of the interval.

## Input:

You will need to read from standard input like so: Mac: `python3 intervals.py < intervals.in`

Windows: `python intervals.py < intervals.in`

The first line in *intervals.in* will be a single positive integer N $(1 < N < 100)$. This number denotes the number of intervals. This will be followed by N lines of data. Each line will have two integer numbers denoting an interval. The first number will be strictly less than the second number. The intervals will not be in sorted order. Assume that the data file is correct. Here is the format of *"intervals.in"*.

```
15
14 17
-8 -5
26 29
-20 -15
12 15
2 3
-10 -7
25 30
2 4
-21 -16
13 18
22 27
-6 -3
3 6
-25 -14
```

You will read in each pair of numbers and create a tuple out of them. You will store the tuples in a list. After you have read all the intervals and the list of tuples is complete, you will call the function *merge_tuples*. This function will return a list of merged tuples in ascending order of the lower end of the tuples.

For the input data file given the function *merge_tuples* will return:

```
[(-25, -14), (-10, -3), (2, 6), (12, 18), (22, 30)]
```

You will take the output of the function *merge_tuples* which is a list of tuples and pass it to the function *sort_by_interval_size()*. This function will return a list of tuples sorted by increasing order of the size of the intervals. If two intervals are of the same size then you should print the two intervals in ascending order of their lower ends.

The list returned will be as follows:

```
[(2, 6), (12, 18), (-10, -3), (22, 30), (-25, -14)]
```

# Output:

You will print your output to standard out. Your ”intervals.out” will have exactly two lines. The first line will be the output of the function *merge_tuples* and the second line will be the output of the function *sort_by_interval_size()*. For the given input, the output will be as follows:

```
[(-25, -14), (-10, -3), (2, 6), (12, 18), (22, 30)]
[(2, 6), (12, 18), (-10, -3), (22, 30), (-25, -14)]
```

## 1.1 Turnin

Turn in your assignment on time on Gradescope system on Canvas. For the due date of the assignments, please see the Gradescope and Canvas systems.

## 1.2 Python Code Style Format

Your code format style should be according to the PEP8 recommendation. Read more about PEP8 here `https://peps.python.org/pep-0008/`

10% of your grade is for code style format and our grading system will reduce points for any violations.

## 1.3 Pair Programming

For this assignment you may work with a partner. Both of you must read the paper on Pair Programming[1] and abide by the ground rules as stated in that paper. If you are working with a partner then only one of you will be submitting the code. But make sure that your partner's name and UT EID is in the header. If you are working alone then remove the partner's name and eid from the header.

## 1.4 Academic Misconduct Regarding Programming

In a programming class like our class, there is sometimes a very fine line between ”cheating” and acceptable and beneficial interaction between students (In different assignment groups). Thus, it is very important that you fully understand what is and what is not allowed in terms of collaboration with your classmates. We want to be 100% precise, so that there can be no confusion.

[1] Read this paper about Pair Programming `https://collaboration.csc.ncsu.edu/laurie/Papers/Kindergarten.PDF`

The rule on collaboration and communication with your classmates is very simple: you cannot transmit or receive code from or to anyone in the class in any way – visually (by showing someone your code), electronically (by emailing, posting, or otherwise sending someone your code), verbally (by reading code to someone) or in any other way we have not yet imagined. Any other collaboration is acceptable.

The rule on collaboration and communication with people who are not your classmates (or your TAs or instructor) is also very simple: it is not allowed in any way, period. This disallows (for example) posting any questions of any nature to programming forums such as **StackOverflow**. As far as going to the web and using Google, we will apply the **”two line rule”**. Go to any web page you like and do any search that you like. But you cannot take more than two lines of code from an external resource and actually include it in your assignment in any form. Note that changing variable names or otherwise transforming or obfuscating code you found on the web does not render the ”two line rule” inapplicable. It is still a violation to obtain more than two lines of code from an external resource and turn it in, whatever you do to those two lines after you first obtain them.

Furthermore, you should cite your sources. Add a comment to your code that includes the URL(s) that you consulted when constructing your solution. This turns out to be very helpful when you’re looking at something you wrote a while ago and you need to remind yourself what you were thinking.

We will use the following Code plagiarism Detection Software to automatically detect plagiarism.

- Staford MOSS

  `https://theory.stanford.edu/~aiken/moss/`

- Jplag - Detecting Software Plagiarism

  `https://github.com/jplag/jplag` and `https://jplag.ipd.kit.edu/`