CSCI 320, Life Beyond Python, Spring 2022
Project 1: The dot product

Project description

One of the most common numerical kernels is the dot product. Given two vectors $(x_1, x_2, \ldots, x_n)$ and $(y_1, y_2, \ldots, y_n)$, their dot product is

$$x_1y_1 + x_2y_2 + \cdot + x_ny_n.$$

Starting with the code

dot.c

```
double dot(const double *const x, const double *const y, const long int n)
{
}
```

write a C function `dot()` that returns the dot product of the vectors `x` and `y`. The latter are pointers to two C arrays of length `n`. Do not change the signature of the function. Be sure to accumulate the answer in double precision.

The `const` declaration in `const long int n` means that `n` cannot be changed in the function. The declaration `const * const x` means that `x` is a constant pointer to constant values. That is, we cannot change `x` nor anything it points to (e.g., `x[0]`).

What to do

Create your C code in a file named `dot.c`. If you want to test it while keeping your test code in a separate file, here is how you do it. If your test code (including `main()`) is in the file `main.c`, then you can build an executable with

```
gcc -Wall -pedantic -o dot main.c dot.c
./dot
```

You can compile your `dot.c` into object code, the first step in building an executable, with the command

```
gcc -Wall -pedantic -c dot.c
./dot
```

This will create a file `dot.o`.

Submit your file `dot.c` and the corresponding Linux object file `dot.o` to the autograder on Gradescope for grading. An object file built on a Mac or a Windows system will **not** work in a Linux environment such as Gradescope.