Not sure exactly what Paris is referring to, but there is a mistake in the provided answers for Q3. The final correct answer for how to calculate the number of comparisons for N locations should state (N-1)(N)/2, not N(N+1)/2. I've flagged this as a problem and made a report - students might lose marks for no good reason!

https://www.coursera.org/learn/computational-thinking-problem-solving/peer/4ylH7/greedy-vs-brute-force-algorithms/discussions/threads/ruWrEuq0EeidKw504uK68g

Hi,

For a sum of numbers going from 1 to n, adding all the numbers equals n*(n+1) / 2 for all natural numbers. This can be proven by induction:

Base Case: We can prove this formula when n = 1. 1 = 1 * (1 + 1) / 2.

Induction Step: Now, we assume the formula works for some n = k and we want to prove it works for n = k + 1. i.e.:

Assume the sum of k is equal to k*(k+1) / 2. Now we want to show that:

1 + 2 + 3 ... + k + k+1 = (k+1)*(k+2) / 2.

Since we assume that 1 through k is k*(k+1) / 2, the sum now becomes:

k*(k+1) / 2 + (k+1).

Combining the terms, we get: k*(k+1) / 2 + 2 (k+1) / 2 = (k+1)(k+2) / 2.

This proves our claim.

Hope this helps!

How computer reads writes to memory:

Let's first look at how the computer writes data to memory.
First, while executing an instruction the control unit determines the address to which to write the data and the value of the data.
The address and data are put on the wires that connect the control unit to memory and they are transferred there.
Third, the memory hardware determines which part of its electrical circuit represents that address and sets the voltage of each of the bits at that address so they hold the data.
What if we want to read from memory?
That is we want to get some data from a particular memory address.

First, while executing an instruction,
the control unit would determine the address from which to read the data.
The address is put on the wires connecting the control unit to memory and
is transferred there.
In this case the memory hardware determines which part of its electrical
circuit represents that address and reads the values of all those bits.
It then puts those values on the wire going back to the control unit
where they're grouped together into the appropriate number of bytes used for
representing that piece of data and can be used by the current instruction.
In the von Neumann architecture, all the instructions and data for
our running programs are stored in memory.
Of course, we need the computer to execute those instructions and modify and
use the data from memory, and that's the job of the CPU.
The CPU contains two parts, the control unit and the arithmetic and
logic unit, or ALU.
Let's talk about the control unit first.
Its job is to get the next instruction to execute, execute it, and
move on to the next one.
The control unit's life is very simple.
It just does these four steps over and over, and over again.
The control unit has a special piece of memory that holds the memory address that
the next instruction to execute.
So first it reads or fetches that instruction,
then it decodes that instruction which means figuring out what it needs to do and
making sure that any input data that it needs is in the right place.
It then executes the instruction and make sure that the output or
results are in the right place.
And, finally, it updates the address of the next instruction to execute and
repeats.
Usually the next instruction to execute is the one that directly follows in memory
the one we just executed.
But in some cases that may change,
as we'll see later in this part of the course.
The control unit also has some temporary memory that is used to hold the operands
and results of operations.
This temporary memory are known as registers.
As part of executing an instruction the control unit uses the arithmetic and
logic unit or ALU to actually perform the mathematical operations.
The ALU contains electronic circuits that can do things like add or
multiply binary numbers and also perform basic logical operations.
Even though what we really want is for the computer to do things like find
the largest number in a collection or determine how similar two dogs are all it
can do is very simple math and logical operations.

But in our programs we just use combinations of those operations to
implement more complicated algorithms as we'll see throughout this part
of the course.
Let's finish up our overview of the von Neumann architecture by looking at input
and output.
When an input device, like a keyboard or mouse, wants to provide data, it sends
a signal to the CPU called an interrupt, to say hey, I have some data for you.
Are you ready for it?
Then the CPU starts running a special program to go get it and
decide what to do with it.
When the CPU wants to send data to an output device like a monitor,
it puts the data some place and then sends a signal to the device saying hey,
I have some data for you, come get it.
And then the output device handles it, depending on what type of hardware it is.
In summary, the hardware for
most modern computers is based on the von Neumann architecture.
This consists of three parts.
Memory, which stores data and instructions.
The CPU which executes instructions and processes data.
And I/O which allows the computer to interact with the outside world.

Quiz von Neumann Architecture:

Consider a collection of values in which individual elements can be accessed using a 0-based index, such that the first element is at index 0, the second is at index 1, and so on. If there are N elements in the collection and the order of the elements is reversed, what is the index in the new (reversed) collection of the element that was at index K in the original collection?

A. N - K

B. N - K + 1

C. N - K - 1

D. (N + K) / 2

E. (N - K) / 2

Correct

Correct. We can solve this via a simple example. Let's say the collection held the values (Ant, Bat, Cow, Dog, Emu, Fox) such that N = 6. The value Emu is at index 4 in this list. If the list were reversed and became (Fox, Emu, Dog, Cow, Bat, Ant), then Emu would be at index 1.Of the options provided, this fits C and E, since N - K - 1 = 6 - 4 - 1= 1 and (N - K) / 2 = (6 - 4) / 2 = 1. Intuitively, we would not think that option E is correct, since there is no need to divide by two when determining the new index, but we can demonstrate that this is incorrect using a different example such as Fox, which is at index 5 in the original and index 0 in the new collection. This fits option C since N - K - 1 = 6 -5 -1 = 0, but not option E since (N - K) / 2 = (6 - 5) / 2 = 1/2 , not 0.

Von Neumann Architecture Peer Graded Assignment:
Q2:

1. READ1 X
2. READ2 Y
3. ADD
4. **WRITE M**
5. **READ1 M**
6. READ2 Z
7. **SUB**
8. **WRITE M**

Q3:

1. READ1 A
2. READ2 A
3. MULT
4. **WRITE T**
5. READ1 T
6. **MULT**
7. WRITE T
8. READ1 B
9. READ2 B
10. MULT
11. **WRITE C**
12. **READ1 T**
13. READ2 C
14. ADD
15. WRITE C

Credits:

All notes from the **Computational Thinking For Problem** Course by University of Pennsylvania on Coursera.