COMP110 WORKSHEET C: COMPUTATIONAL COMPLEXITY

Version 1.0
BSc Computing for Games

Dr Ed Powley

Tasks

This worksheet tests your understanding of the concept of computational complexity, as well as your ability to communicate this understanding in writing.

Consider the following algorithm:

```
1:  procedure HasDuplicate(list)
2:      let n be the length of list
3:      for i = 0, 1, ..., n − 1 do
4:          for j = 0, 1, ..., n − 1 do
5:              if i ≠ j and list[i] = list[j] then
6:                  return true
7:              end if
8:          end for
9:      end for
10:     return false
11: end procedure
```

(a) **State** what task this algorithm performs.
(b) **Explain** why the worst case running time of the algorithm is quadratic, i.e. $O(n^2)$.

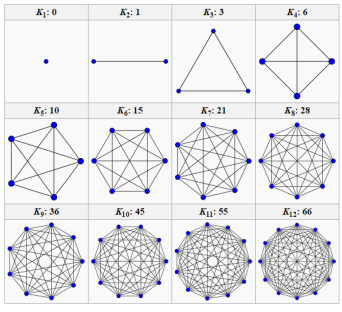
Now suppose that the **for** loop on line 4 is changed so that j ranges from 0 to $i - 1$.

(c) **Explain** why the algorithm is still correct.
(d) **Explain** why the algorithm will run approximately twice as fast.
(e) Is the time complexity of the algorithm still quadratic? **Explain** your answer.

Now consider the following algorithm, which performs the same task as the first:

```
1:  procedure HasDuplicate(list)
2:      let n be the length of list
3:      sortedList ← sort(list)
4:      for i = 1, 2, ..., n − 1 do
5:          if list[i − 1] = list[i] then
6:              return true
7:          end if
8:      end for
9:      return false
10: end procedure
```

(f) With reference to an appropriate source, **write down** the time complexity of Python's built-in `sort` function, in big-O notation.
(g) Thus **write down** the time complexity of the above algorithm in big-O notation. **Explain** your answer.
(h) If the size of the input list is large, which of these two algorithms is likely to run faster? **Explain** your answer.
(i) **Suggest one** reason why a programmer might choose the "slower" algorithm over the "faster" one.



Considering every pair of elements in a data structure often leads to quadratic complexity.

Submission instructions

If you did not already do so for a previous worksheet, **fork** the GitHub repository at the following URL:

`https://github.com/Falmouth-Games-Academy/comp110-worksheets`

Within the `worksheet_C` directory (which you should create if it does not exist), write your answers to questions (a)–(i) in the `README.md` file. Open a **pull request**.

Attend the scheduled worksheet feedback session on **Monday October 24th 2016**, ensuring that you have uploaded all material to GitHub and opened a pull request before this time.

Marking criteria

Remember that **it is better to submit incomplete work than to submit nothing at all**. Any attempt, even unfinished, will receive a passing grade.

Your work will be marked according to the following criteria:

- Where appropriate, are your answers **correct**?
- Are your explanations **clear**, **concise** and **accurate**?
- Where you have obtained information from external sources, are they **properly cited**?