ASSIGNMENT 1: COMPLEXITY ANALYSIS

due date: Sunday, 27 February 2022 (23.59 WITA)

Aturan pengerjaan tugas:

1. Tugas boleh diketik/ditulis tangan (pastikan bisa dibaca), boleh menggunakan Bahasa Indonesia/Inggris. Hindari menggunakan tinta merah.
2. Tugas dikumpulkan dalam format pdf. Jika menggunakan tulis tangan, harap discan (tidak difoto), kemudian dikompresi untuk memperkecil ukuran file.
3. Format penamaan tugas: **DAA01_Kelas_Nama Lengkap_NIM**.
   Contoh: **DAA01_6A_Gede Ganesha_1610101001**
4. Pengumpulan tugas melalui e-learning Undiksha.
5. Anda diizinkan untuk berdiskusi dengan rekan Anda. Namun Anda harus menuliskan/menjelaskan jawaban Anda sendiri, dan paham dengan baik apa yang Anda tulis. Anda siap bertanggung jawab terhadap hasil pekerjaan Anda. Hasil pekerjaan yang memiliki kemiripan yang tinggi dengan pekerjaan mahasiswa lain mempengaruhi poin penilain.
6. Tugas dinilai berdasarkan kejelasan serta kesesuaian jawaban/penjelasan dengan pertanyaan yang diajukan. Total nilai maksimum tugas ini adalah 115. Keterlambatan dalam pengumpulan tugas mengurangi poin penilaian.

Dengan ini, Anda menyatakan bahwa Anda siap menerima segala konsekuensi jika nantinya ditemukan adanya kecurangan dalam pengerjaan tugas ini.

Problems

1. (*Computing time complexity*, **15 point**)

    1. What is the time complexity of the "naive gcd algorithm" and the "Euclidean algorithm" explained in the lecture? Justify your answer!

Algorithm 1 Naive gcd algorithm of two integers

```
1: procedure GCD(m, n)
2:     r = 1
3:     x = min(m, n)
4:     for i = 1 to x do
5:         if a MOD i == 0 and b MOD i == 0 then r = i
6:         end if
7:     end for
8: end procedure
```

Algorithm 2 Euclidean algorithm

```
1: procedure EUCLIDGCD(m, n)
2:     while b ≠ 0 do
3:         r = a MOD b
4:         a = b
5:         b = r
6:     end while
7:     return a
8: end procedure
```

    2. Compute the best-case, worst-case, and average-case complexities of the following algorithm!

Algorithm 3 Sequential search

```
1: procedure SEQSEARCH(A[1..n], x)
2:     found ← False
3:     i ← 1
4:     while (not found) and (i ≤ N) do
5:         if (A[i] = x) then found ← True
6:         else i ← i + 1
7:         end if
8:     end while
9:     if (found) then index ← i
10:    else index ← 0
11:    end if
12: end procedure
```

2. (*Dominant terms in a function*, **20 point**)

Select the dominant term(s) and specify the lowest $\mathcal{O}$-complexity of each algorithm! Give a short proof or an explanation for each function to justify your answer!

	Expression	**Dominant term(s)**	$\mathcal{O}(\cdot)$
1.	$5 + 0.001n^3 + 0.025n$		
2.	$500n + 100n^{1.5} + 50n \log_{10} n$		
3.	$0.3n + 5n^{1.5} + 2.5 \cdot n^{1.75}$		
4.	$n^2 \log_2 n + n(\log_2 n)^2$		
5.	$n \log_3 n + n \log_2 n$		
6.	$3 \log_8 n + \log_2 \log_2 \log_2 n$		
7.	$2n + n^{0.5} + 0.5n^{1.25}$		
8.	$0.01n \log_2 n + n(\log_2 n)^2$		
9.	$100n \log_3 n + n^3 + 100n$		
10.	$0.003 \log_4 n + \log_2 \log_2 n$		

3. (*Comparing time complexities of two algorithms*, **20 point**)

   (a) Suatu algoritma memiliki kompleksitas waktu $\mathcal{O}(f(n))$ dan *running time* $T(n) = cf(n)$. Jika algoritma tersebut membutuhkan waktu 10 detik untuk memproses 1000 data, berapakah waktu yang dibutuhkan untuk memproses 100,000 data jika $f(n) = n$? Bagaimana halnya jika $f(n) = n^3$?

   (b) Algoritma **A** dan **B** masing-masing memiliki kompleksitas waktu $T_A(n) = 5n \log 10n$ and $T_B(n) = 25n$ microdetik, jika diberikan input dengan ukuran n. Dalam aturan Big-O, algoritma manakah yang bekerja lebih baik? Berikan batasan nilai n agar algoritma **B** memiliki kompleksitas waktu yang lebih baik!

   (c) Suatu perangkat lunak **A** dan **B** masing-masing memiliki kompleksitas waktu $\mathcal{O}(n \log n)$ dan $\mathcal{O}(n^2)$. Keduanya memiliki running time $T_A(n) = c_A n \log_{10} n$ dan $T_B(n) = c_B n$ milidetik untuk memproses data dengan ukuran n. Selama proses pengujian, waktu rata-rata pemrosesan $n = 10^4$ data dengan perangkat **A** dan **B** masing-masing adalah 100 milidetik dan 500 detik. Berikan batasan nilai n agar perangkat **A** bekerja lebh cepat dibandingkan perangkat **B**, dan berikan batasan nilai n untuk kondisi sebaliknya. Perangkat mana yang Anda rekomendasikan jika kita harus memproses data dengan ukuran tidak lebih dari $n = 10^9$?

4. (*Formal proof*, **20 point**)

   (a) Using the formal definition of Big-O notation, prove the following:

   $$T(n) = a_0 + a_1 n + a_2 n^2 + a_3 n^3 \in \mathcal{O}(n^3)$$

   Hint: temukan konstanta c dan batasan nilai n_0 sedemikian sehingga $cn^3 \geq T(n)$ untuk $n \geq n_0$.

   (b) Let $T_1(n) \in \mathcal{O}(f(n))$ and $T_2(n) \in \mathcal{O}(g(n))$. Prove the following:

   (a) $T_1(n) + T_2(n) \in \mathcal{O}(f(n)) + \mathcal{O}(g(n)) \in \mathcal{O}(\max(f(n), g(n)))$
   (b) $T_1(n) \cdot T_2(n) \in \mathcal{O}(f(n)) \cdot \mathcal{O}(g(n)) \in \mathcal{O}(f(n) \cdot g(n))$
   (c) $\mathcal{O}(cf(n)) \in \mathcal{O}(f(n))$, where c is a constant
   (d) $f(n) \in \mathcal{O}(f(n))$

5. (*Big-$\mathcal{O}$, Big-Ω, and Big-Θ*, **5 point / question, total 40 point**)

   Let $f, g : \mathbb{N}^+ \to \mathbb{N}^+$ be two functions. Prove or invalidate each of the following:

   1. If $f(n) \in \mathcal{O}(g(n))$ then $g(n) \in \mathcal{O}(f(n)$
   2. $f(n) + g(n) \in \Theta(\min(f(n), g(n)))$ (assuming that it exists)
   3. If $f(n) \in \mathcal{O}(g(n)$, then $\log(f(n)) \in \mathcal{O}(\log(g(n))$
   4. If $f(n) \in \mathcal{O}(g(n))$ then $2^{f(n)} \in \mathcal{O}(2^{g(n)})$
   5. $f(n) \in \mathcal{O}(f^2(n))$
   6. If $f(n) \in \mathcal{O}(g(n))$ then $g(n) \in \Omega(f(n))$
   7. $f(n) \in \Theta(f(n/2))$
   8. If $g(n) \in \mathcal{O}(f(n)$ then $f(n) + g(n) \in \Theta(f(n))$