Lembar Kerja Responsi 2
Mata Kuliah KOM 401 Analisis Algoritme
Semester Ganjil Tahun Akademik 2020/2021

Asisten Praktikum:

1. **Alfian Hamam Akbar**
2. **Hilmi Farhan Ramadhani**

---

1. $f(n)$ sangat membantu kita memahami seberapa baik suatu algoritme. Namun, pada prakteknya f(n) dapat disederhanakan. Oleh karena itu, sederhanakan $f(n)$ berikut:
   a. $f(n) = 6n^2 + 2n - 8$
   b. $f(n) = n(9n^3 - n)$
   c. $\frac{f(n) = n(n+1)(n+2)}{2}$
   d. $f(n) = n(\frac{1}{2}n^2 + n + 1)$
   a. $f(n) = n^2$
   b. $f(n) = n^4$
   c. $f(n) = n^3$
   d. $f(n) = n^3$
2. Tahun "semi-kabisat" adalah tahun yang bukan merupakan tahun kabisat, tetapi jika tiap bilangan penyusun angka tahunnya dijumlahkan akan habis dibagi dengan 4. Ada berapa tahun "semi-kabisat" semenjak tahun 1901 hingga 1960?
   Pada tahun 19<u>01</u> sampai 19<u>60</u> hanya dua digit terakhir pada angka tahun yang berubah. Karena sisa pembagian dari jumlah dua digit pertama adalah 2 ((1+9) mod 4), maka jumlah dua digit terakhir harus memiliki sisa pembagian 2 jika dibagi dengan 4.
   Banyak bilangan dari 1 - 60 yang jumlah digitnya memiliki 2 sebagai sisa pembagian
   190* : 1902, 1906
   191* : 1911, 1915, 1919
   192* : 1920, 1924, 1928
   193* : 1933, 1937
   194* : 1942, 1946
   195* : 1951, 1955, 1959
   196* : 1960

   1920, 1924, 1928, dan 1960 merupakan tahun kabisat, jadi jumlah tahun "semi-kabisat" : 12.

3. Urutkan kompleksitas persamaan di bawah ini dari yang terkecil
   a. n^n
   b. $n + 1$
   c. $n^3 - 2$
   d. $log(n)$
   e. 2^n
   f. 98
   g. $n!$
   h. $\sqrt{n} + 2$

   f, d, h, b, c, e, g, a

4. Buktikan dengan kontradiksi untuk setiap a, b bilangan bulat, $a^2 - 4b - 2 \neq 0$

$a^2 - 4b - 2 = 0 \Rightarrow a^2 = 4b + 2 \Rightarrow a^2 = 2(2b + 1)$,
jika kita asumsikan $2b + 1 = m$, maka $a^2 = 2(2b + 1) = 2m$

Terlihat bahwa a^2 adalah bilangan genap sehingga a juga bilangan genap.

Karena a bilangan genap maka kita asumsikan a = 2k , dengan k bilangan bulat. Sehingga persamaannya menjadi :

$a^2 = 2(2b + 1) \Rightarrow (2k)^2 = 2(2b + 1)$
$\Rightarrow 4k^2 = 2(2b + 1) \Rightarrow 2k^2 = 2b + 1 \Rightarrow 2k^2 - 2b = 1$
$2k^2 - 2b = 1 \Rightarrow 2(k^2 - b) = 1$,
Kita asumsikan $k^2 - b = n$ sehingga persamaannya menjadi :
$2(k^2 - b) = 1 \Rightarrow 2n = 1$

Dari sini kita dapatkan bahwa 1 adalah bilangan genap sedangkan kenyataannya 1 adalah bilangan ganjil sehingga terdapat sebuah kontradiksi.
Asumsi kita salah.
Sehingga terbukti bahwa **untuk setiap a, b bilangan bulat, $a^2 - 4b - 2 \neq 0$.**

5. Buktikan dengan induksi matematika bahwa:

$$P(n) : 1^2 + 2^2 + 3^2 + ... + n^2 = (\tfrac{n}{2}(n + 1))^2$$

Sorry gais salah soal ya hehe, ini yang bawah yang bener

$$P(n) : 1^3 + 2^3 + 3^3 + ... + n^3 = (\tfrac{n}{2}(n + 1))^2$$

Langkah 1

Akan ditunjukkan P(1) benar.

$$P(1) : 1^3 = \left(\frac{1}{2}(1+1)\right)^2$$

$$P(1) : 1^3 = 1 \ldots (benar)$$

Langkah 2

Asumsikan P(k) benar, sehingga

$$1^3 + 2^3 + 3^3 + \cdots + k^3 = \left(\frac{k}{2}(k+1)\right)^2$$

akan ditunjukkan P(k+1) juga benar, yaitu

$$1^3 + 2^3 + 3^3 + \cdots + k^3 + (k+1)^3 = \left(\frac{(k+1)}{2}((k+1)+1)\right)^2$$

$$1^3 + 2^3 + 3^3 + \cdots + k^3 + (k+1)^3 = \left(\frac{k}{2}(k+1)\right)^2 + (k+1)^3$$

$$1^3 + 2^3 + 3^3 + \cdots + k^3 + (k+1)^3 = \frac{k^2}{4}\ (k+1)^2 + (k+1)^3$$

$$1^3 + 2^3 + 3^3 + \cdots + k^3 + (k+1)^3 = (k+1)^2 \left(\frac{k^2}{4} + (k+1)\right)$$

$$1^3 + 2^3 + 3^3 + \cdots + k^3 + (k+1)^3 = (k+1)^2 \left(\frac{k^2 + 4k + 1}{4}\right)$$

$$1^3 + 2^3 + 3^3 + \cdots + k^3 + (k+1)^3 = \frac{(k+1)^2}{4}(k+2)^2$$

$$1^3 + 2^3 + 3^3 + \cdots + k^3 + (k+1)^3 = \left(\frac{(k+1)}{2}((k+1)+1)\right)^2$$

Kalo jawaban LKP kemarin **TIDAK TERBUKTI,** kalo diatas jawaban yang soal setelah dibenarin

6. Tentukan kompleksitas dari algoritme berikut:

   a. Metode 1

```
void method1(int [] arr)
{
        int n = arr.length;
        for(int i = n - 1 ; i >= 0;  i = i - 3)
        {
                SOP (arr[i]);
        }
}

        Remark: We're assuming that SOP (System.out.println) is O(1).
```

O(n)

   b. Metode 2

```
void method2(int [] arr)
{
        for(int i = 0; i <arr.length; i++)
        {
                for(int k = 0; k <arr.length; ++k)
                {
                        SOP (Math.log(arr[i]));
                }
        }
}
```

$O(n^2)$

   c. Metode 3

```
void method3(int [] arr) {
        for(int i = 0; i <arr.length; i++)
        {
                method1(arr);
                method2(arr);
        }
}
```

$O(n^3)$

d. Metode 4

```
void method4(int [] arr)
{
        for(int i = 0; i <arr.length; i++)
        {
                for(int k = arr.length - 1;  k > 0; k = k / 3 )
                {
                        SOP (arr[i]);
                }
        }
}
```

O(n log n)

7. Efisiensi suatu algoritma dapat diukur dengan menghitung cost yang dilihat dari operasi operasi dasar yang dijalankan dalam suatu algoritme. Hitunglahlah total cost beberapa potongan program dibawah ini

a	`min = 0`
	`max = 0`
	`while(n != 0){`
	`scanf("%d",&num)`
	`if(num >= max)`
	`max = num`
	`else if(num <= min)`
	`min = num`
	`printf("%d %d\n",min,max)`
	`n = n - 1`
	`}`
		c1
		c2
		c3
		c4
		c5
		c6
		c7
		c8
		c9
		c10
			1
			1
			n+1
			n
			n
			n
			n
			n
			n
			n
				Total cost:
				c1 + c2 + (n+1)*c3 + n*c4 + n*c5 + n*c7 + max(c6,c8) + n*c9 + n*c10

b	`for(int i=0; i<n; i++){`
	`for(int j=0; j<m; j++){`
	`printf("%d",a[i][j]);`
	`if(j==m-1)`
	`printf("\n");`
	`else`
	`printf(" ");`
	`}`
	`}`
		c1
		c2
		c3
		c4
		c5
		c6
			n + 1
			n(m+1)
			nm
			nm
			nm
			nm
				Total cost:
				(n+1)*c1 + n(m+1)*c2 + nm*c3 + nm*c4 + nm*max(c5,c6)

c	`for(int i=0; i<n; i++){`
	`scanf("%d",&n);`
	`for(int j=0; j<n; j++){`
	`scanf("%d",&a[j]);`
	`}`
	`for(int j=0; j<n; j++){`
	`if(a[j+1]>a[j])`
	`count++;`
	`else if (a[j+1]<a[j])`
	`count++;`
	`}`
	`if(count<=2)`
	`printf("Yes\n");`
	`else`
	`printf("No\n");`
	`count=0;`
	`}`
		c1
		c2
		c3
		c4
		c5
		c6
		c7
		c8
		c9
		c10
		c11
		c12
		c13
			n+1
			n
			n(n+1)
			n^2
			n(n+1)
			n^2
			n^2
			n^2
			n^2
			n
			n
			n
			n
				Total cost:
				(n+1)*c1 + n*c2 + n(n+1)*c3 + n^2*c4 + n(n+1)*c5 + n^2*c6 + n^2*c8 + n^2*max(c7,c9) + n*c10 + n*max(c11,c12) + n*c13

8. Tentukan nilai grow rate dari fungsi berikut jika memproses n data sebesar 4, 16, 64, 256, 1024
   a. $log(n)$
   b. $n\ log(n)$
   c. $\sqrt{n}$
   d. n^3
   e. 2^n
   f. $n!$

	4	16	64	256	1024
$log(n)$	2	4	6	8	10
$n\ log(n)$	8	64	384	2048	10240
$\sqrt{n}$	2	4	8	16	32
n^3	64	4096	262144	16777216	1073741824
2^n	16	65,536	1.84 * 10^19	1.15 * 10^77	1.79 * 10^308
$n!$	24	2.09 * 10^13	1.26 * 10^89	Infinity	Infinity