SOAL UJIAN TENGAH SEMESTER GANJIL TAHUN AJARAN 2024/2025 PROGRAM STUDI TEKNIK INFORMATIKA UNIVERSITAS KOMPUTER INDONESIA	
Mata Kuliah : Analisis dan Strategi Algoritma Hari / Tanggal : Nama : Idin Naufal Hakim NIM : 10123157 Kelas : IF4	Waktu / Sifat : / OPEN BOOK Semester : III/ Ganjil Dosen : TIM DOSEN

Tata cara pengerjaan:

1. Soal UTS ini adalah salah satu dari komponen penilaian untuk mata kuliah Analisis dan Strategi Algoritma dan mengetahui pemahaman mahasiswa terkait materi yang sudah disampaikan oleh dosen pengampu.
2. Jawaban dikumpulkan paling lambat hari Jumat, 06 Desember 2024 jam 13.40.
3. Jawaban UTS dikirimkan ke link berikut ini dengan format nama file (**nama_nim.pdf**) KELAS IF-4: https://drive.google.com/drive/folders/1299Fs14MrZGy9Ygh7CpdTJjbdC4SnTz?usp=sharing

   KELAS IF-5:
   https://drive.google.com/drive/folders/1rz1JtCLHhFBGI9EqnvS2JH0SgS7w4DIH?usp=sharing

   KELAS IF-6:
   https://drive.google.com/drive/folders/1A3WTE_BTEUKm_ybvBK7PnF4WWjlF5KE?usp=sharing

   KELAS IF-7:
   https://drive.google.com/drive/folders/17AhPfFVBGN6M3p4zYbFGvrDWTJgmIAbT?usp=sharing

SOAL UTS

1. Buatlah algoritma dalam bentuk pseudo code untuk mengecek apakah 10 deret pertama bilangan bulat termasuk bilangan genap. Setelah algoritma dibuat, hitung kompleksitas waktu asimptotik dari algoritma tersebut!

   Jawab:

   Kamus:

   ```
   i : integer
   ```

   Algoritma:

   ```
   for i <- 1 to 10 do
     if i mod 2 = 0
      then
        output i + "adalah genap"
     else
        output i + "adalah ganjil"
   ```

```
    endif
  endfor
```

Algoritma di atas memiliki satu loop for yang berjalan dari 1 hingga 10, loop ini akan dieksekusi sebanyak 10 kali.

Di dalam loop, terdapat pernyataan if yang memeriksa apakah i adalah genap atau ganjil. Operasi ini melibatkan satu operasi modulus (i mod 2) dan satu perbandingan. Ini adalah operasi konstan, yaitu O(1).

Dalam notasi asimptotik, kita biasanya mengabaikan konstanta dan fokus pada pertumbuhan fungsi. Oleh karena itu, meskipun kita memiliki 10 iterasi, kompleksitas waktu asimptotik dari algoritma ini adalah O(1)

Kompleksitas waktu asimptotik dari algoritma ini adalah O(1), karena jumlah iterasi (10) adalah tetap dan tidak tergantung pada ukuran input yang lebih besar

2. Berapa kompleksitas waktu asimptotik yang dibutuhkan pada penggalan algoritma tersebut (gunakan teorema dan aturan perhitungan yang sudah dijelaskan sebelumnya)? Jelaskan!

```
int i = 1;
    int j;  while (i
<= n) {    j = 1;
while (j <= i) {
    printf("%i * %i = %i\n",i,j,j*i);
j++;    }    i++;
  }
```

Jawab:

1. Loop Luar (while (i <= n)):

Variabel i dimulai dari 1 hingga n

.Jadi, loop luar dijalankan n kali.

2. Loop Dalam (while (j <= i)) :

Untuk setiap nilai i variabel j dimulai dari 1 hingga i.

Jadi, jumlah iterasi dalam loop ini adalah i

3. Operasi Dasar:

Dalam loop terdalam, hanya ada satu operasi dasar, yaitu printf, yang dieksekusi sekali untuk setiap iterasi loop dalam.

Total Operasi:

Untuk setiap iterasi loop luar (i dari 1 hingga n):

Loop dalam berjalan i kali.

Total operasi dapat dinyatakan sebagai jumlah total iterasi dari semua loop dalam perhitungan total operasi menggunakan rumus aritmetika :

$$\frac{n(n+1)}{2}$$

ketika n menjadi sangat besar, suku dominan adalah n^2 / 2, sehingga kompleksitas waktu asimptotiknya adalah : O (n^2)

3. Apa output dari algoritma tersebut apabila n diisi 10? Jelaskan!

Jawab:

1 * 1 = 1
2 * 1 = 2
2 * 2 = 4
3 * 1 = 3
3 * 2 = 6
3 * 3 = 9
4 * 1 = 4
4 * 2 = 8
4 * 3 = 12
4 * 4 = 16
5 * 1 = 5
5 * 2 = 10
5 * 3 = 15
5 * 4 = 20
5 * 5 = 25
6 * 1 = 6
6 * 2 = 12
6 * 3 = 18
6 * 4 = 24
6 * 5 = 30
6 * 6 = 36
7 * 1 = 7
7 * 2 = 14
7 * 3 = 21
7 * 4 = 28

7 * 5 = 35
7 * 6 = 42
7 * 7 = 49
8 * 1 = 8
8 * 2 = 16
8 * 3 = 24
8 * 4 = 32
8 * 5 = 40
8 * 6 = 48
8 * 7 = 56
8 * 8 = 64
9 * 1 = 9
9 * 2 = 18
9 * 3 = 27
9 * 4 = 36
9 * 5 = 45
9 * 6 = 54
9 * 7 = 63
9 * 8 = 72
9 * 9 = 81
10 * 1 = 10
10 * 2 = 20
10 * 3 = 30
10 * 4 = 40
10 * 5 = 50
10 * 6 = 60
10 * 7 = 70
10 * 8 = 80
10 * 9 = 90
10 * 10 = 100

Algoritma mencetak tabel perkalian hingga n.
Loop luar mengontrol angka pertama (i)
Loop dalam mencetak semua perkalian dari j = 1 hingga j = i untuk setiap nilai i.

4. Hitung T(n) dan Big Oh (*O*) dari penggalan algoritma di atas!
   Jawab:

   T(n) = n^2

   Big O :
   T(n) <= c.f(n)
   n^2 <= c. n^2
   $n^2 - cn^2 <= 0$
   n^2 (1 – c) <= 0
   n^2 = 0 -> n_0 = 0
   1 – c = 0 -> c = 1
   Maka n_0 = 0 dan c = 1

5. Diberikan penggalan algoritma sebagai berikut:

```
for i := n downto 1 do
begin
   j:=1
   while j ≤ i do
   begin
      output(i)
      bil[j] := bil[j] + i
      j := j + 1
   end
end
```

a) Apa output dari algoritma tersebut apabila n diisi 10? Jelaskan!

10 10 10 10 10 10 10 10 10 10
9 9 9 9 9 9 9 9 9
8 8 8 8 8 8 8 8
7 7 7 7 7 7 7
6 6 6 6 6 6
5 5 5 5 5
4 4 4 4
3 3 3
2 2
1

Loop Luar (for i := n downto 1):
Variabel i dimulai dari n dan menurun ke 1.
Pada setiap iterasi, i adalah angka yang akan digunakan dalam operasi dan output.

Loop Dalam (while j ≤ i):
Untuk setiap nilai i ,j dimulai dari 1 hingga i
Operasi dilakukan sebanyak i kali untuk setiap iterasi loop luar.

Operasi
output(i) : Mencetak nilai i di setiap iterasi loop dalam
bil[j] := bil[j] + i : Menambahkan nilai i ke elemen array bil[j]

b) Hitung T(n) dan Big Oh (*O*) dari penggalan algoritma di atas!
T(n) = n (n + 1) / 2 = n^2 + n /2
Big O

c) Cari 2 contoh algoritma lain yang memiliki Big Oh yang sama dengan algoritma di atas! Kemudian tuliskan algoritmanya dan lakukan perhitungan Big Oh pada kedua algoritma tersebut!

Contoh : Bubble Sort (1)

```
for i from 1 to n do
    for j from 1 to n-i do
        if arr[j] > arr[j+1] then
            swap(arr[j], arr[j+1])
```

Big O: O(n2)

Contoh 2:

```
n := length(arr)

for i := 2 to n do
  key := arr[i]
  j := i - 1

  while j > 0 and arr[j] > key do
    arr[j + 1] := arr[j]
    j := j - 1
    arr[j + 1] := key
```

Big O: T(n)=O(n^2).