Öğrenci Numarası : ______________________ Adı Soyadı : ______________________

Soru	1	2	3	4	5	6	Toplam
Puan	10	15	20	25	30	30	130
Not							

UYARI: 5 ve 6. sorulardan sadece birini çözünüz çözmediğiniz soruyu puan tablosunda çarpı ile işaretleyiniz!

$$\sum_{i=0}^{n} a^i = 1 + a + \cdots + a^n = \frac{a^{n+1}-1}{a-1}\ (a \neq 1); \quad \sum_{i=0}^{n} 2^i = 2^{n+1} - 1$$

1. Aşağıdaki önermelerin doğru olup olmadığını açıklayarak belirleyiniz.
   (a) (2P) $n(n+1) \notin O(n^4)$
   (b) (2P) $2^{n-1} + n \in \Omega(2^n)$
   (c) (2P) $n(n+1) \in \Theta(n^3)$
   (d) (2P) $9n^4 - 7n^3 + 3 \in \Theta(n^4)$
   (e) (2P) $\Theta(\alpha g(n)) \in \Theta(g(n)), \alpha \in N^+$

a. Yanlış → $n^2+2n \leq n^4$
$n \geq 2$ $c=1$ için

b. Doğru → $\frac{1}{4} 2^n \leq 2^{n-1} + n$
$n \geq 1$ $c = 1/4$ için

c. $n(n+1) \in \Theta(n^3)$ yanlış

$n^3 \leq n^2+n \leq 2n^3$
$c_2 = 2$
$n \geq 2$ ama
$n^2 \leq n+1$
$n^2 - n \leq 1$
$n(n-1) \leq 1$ X

d. $9n^4 - 7n^3 + 3 \in \Theta(n^4)$ doğru

$n^4 < 9n^4 - 7n^3 + 3 \leq 10 n^3$
$c_1 = 1$ $9^4 - 7^3$ $c_2 = 10$
$n \geq 1$ $n \geq 1$

e. Doğru $\Theta(\alpha g(n)) \in \Theta(g(n))$ $\alpha \in N^+$

2. Aşağıda verilen fonksiyonların orderlarını (verimlilik sınıflarını) Θ cinsinden veriniz (En basit verimlilik sınıfı fonksiyonunu veriniz.)
   (a) (3P) $2lg(n+65)^6$

$= 12 \lg n (n+65)$
$n \geq 65$
$12 \lg n \leq 12 \lg n (n+65) \leq 24 \lg n$
$12 \lg n \leq 12 \lg (n+65) \leq 24 \lg n$
$\in \Theta(\lg n)$
C1 C2

(b) (3P) $\frac{n(n-1)(2n+5)}{6}$

$$\lim_{n\to\infty} \frac{(n^2-n)(2n+5)}{6n^3} = \lim_{n\to\infty} \frac{2n^3+5n^2-2n^2-5n}{6n^3}$$

$$= \lim \underbrace{\frac{1}{3}}_{} + \underbrace{0\ 0}_{0} = 1/3 \in \Theta(n^3)$$

(c) (3P) $\sqrt{8n^8 + 8n^2 + \sqrt{n}}$

say n^4

$$\lim_{n\to\infty} \sqrt{\frac{8n^8+8n^2+\sqrt{n}}{n^8}}$$

$$\sqrt{8 + \underbrace{\frac{8}{n^6} + \frac{1}{n^{15/2}}}_{}}\ 0 = 2\sqrt{2} \to c$$

$\in \Theta(n^4)$

(d) (3P) $2nlg(2n+2)^3 + (n^2+2)^2 lgn$

$$\underline{6n\lg(2n+2)} + \underline{(n^4+4n^2+4)\lg n} \to n^4 \lg n$$

$$\lim_{n\to\infty} \frac{n^4+4n^2+4}{n^4} = 1$$

(e) (3P) $8^n + 9^{2^n}$

$\in (9^{2^n})$

$$\underbrace{1}_{c_1} 9^{2^n} \le 8^n + 9^{2^n} \le \underbrace{2}_{c_2} 9^{2^n}$$

3. Aşağıda verilen tekrar etme ilişkilerini çözün.

(a) (10P) $T(n) = 2T(n/2) - 2, T(1) = 8$

$T(n/2) = 2T(n/4) - 2$ $\qquad n/2^i = 1$

$T(n/4) = 2T(n/8) - 2$ $\qquad i = \log_2 n$

$T(n/2) = 2(2T(n/8) - 2) - 2$ $\qquad 2^{\log_2 n} 8 - 2n$

$= 2^2 T(n/8) - 4 - 2$ $\qquad 8n$

$T(n) = 2(2^2 T(n/8) - 4 - 2) - 2$ $\qquad 6n + 2n \qquad \in \Theta(n)$

$= 2^3 T(n/8) - 8 - 4 - 2$

$n/16$ $= 2^i T(n/2^i) - 2^i - 2^{i-1} \cdots - 2^1$

$-(2^{i+1} - 2)$

(b) (10P) $T(n) = 2T(n-3) + n, T(1) = 1$

$T(n-3) = 2T(n-6) + n-3$

$T(n-6) = 2T(n-8) + n-6$

$T(n-8) = 2T(n-12) + n-8$

$T(n-3) = 2(2T(n-8) + (n-6)) + n-3$

$= 2^2 T(n-8) + 2(n-6) + n-3$

$= 2^2(2T(n-12) + n-8) + 2(n-6) + (n-3)$

$= 2^3 T(n-12) + 4(n-8) + 2(n-6) + (n-3)$

$T(n) = 2(2^3 T(n-12) + 4(n-8) + 2(n-6) + n-3) + n$

$= 2^4 T(n-12) + 8(n-8) + 4(n-6) + 2(n-3) + n$

$= 2^i T(n-3i) + \sum_{t=0}^{i-1} 2^t (n-3t)$ lower

$i = \frac{n-1}{3}$

$= 2^{\frac{n-1}{3}} \cdot 1 + n(2^{\frac{n-1}{3}} + 1) \leq$ lower

$\left(\frac{n-1}{3} - 1\right) 2^{(\frac{n-1}{3}+1)}$

$\Downarrow$

$\in \Theta(n 2^{n/3})$

4. Algoritma 1'i inceleyiniz.

   (a) (15P) Aşağıda verilen Algoritma 1'in çalışma zamanını bulunuz.

   (b) (10P) Bu algoritmadaki verimsizlik nerden kaynaklanmaktadır? Nasıl düzeltilebilir?

Algorithm 1 Gaussian Elimination

function GE($A[0..n-1, 0..n]$)
  Giriş olarak n x (n+1) boyutunda reel elemanlı bir matris veriliyor
    for $i \leftarrow 0$ `to` $n-2$ **do**
      for $j \leftarrow i+1$ `to` $n-1$ **do**
        for $k \leftarrow i$ `to` n **do** $A[j,k] \leftarrow A[j,k] - A[i,k] * A[j,i]/A[i,i]$

verimsizlik

burada hesaplanabilir

$a^{\log_b x} = x^{\log_b a}$

$$\sum_{i=0}^{n-2} \sum_{j=i+1}^{n-1} \sum_{k=i}^{n} 1 = \sum_{i=0}^{n-2} \sum_{j=i+1}^{n-1} (n-i+1) \qquad \sum_{i=1}^{n} i^2 = \frac{n(n+1)(2n+1)}{6}$$

$$= \sum_{i=0}^{n-2} (n-1-i)(n-i+1)$$

$$\downarrow$$

$$(n-1)(n+1) \quad (n-2)(n) \quad (n-3)(n-1)$$

$$\sum_{z=1}^{n-1} z.(z+2) \quad 1.(3) \quad = \sum_{z=1}^{n-1} z^2+2z = \sum_{z=1}^{n-1} z^2 + 2\sum_{z=1}^{n-1} z \Rightarrow \in \Theta(n^3)$$

$$= \frac{(n-1)n(2n-1)}{6} + n(n-1) + \frac{\cancel{2}(n-1)n}{\cancel{2}}$$

$$= \frac{(n-1)n(2n-1+6)}{6} = \frac{(n-1)n(2n+5)}{6}$$

5. n adet evin düz bir yolda sırasıyla $x_1 < x_2 < ... < x_n$, $x_i \in R^+$ koordinatında bulunduğunu düşünelim. Bu evlerden birine postane açılması istenmektedir.

   (a) (20P) Postanenin nereye açılması gerektiğini bulmak için diğer evler ve açılacak postane arasındaki ortalama uzaklığı en aza indirecek bir algoritma yazınız.

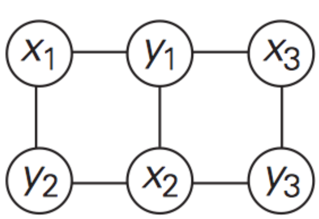
```
Algorithm whereTo (X[1..n])
ort = Xn.n
where = -1
for i=1 to n
    newOrt=0
    for j=1 to n
        newOrt += |Xj - Xi|
    if newOrt < ort
        ort = newOrt
        where = i

return where
```

(b) (10P) Postanenin nereye açılması gerektiğini bulmak postane ile en uzak ev arasındaki mesafeyi minimize eden bir algoritma tasarlayınız.

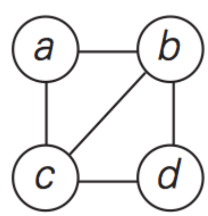
Algorithm whereTo (X[1...n])
max := Xn , where := -1
for i = 1 to n
if max(|X1 - Xi|, |Xn - Xi|) < maxi
maxi := max(|X1 - Xi|, |Xn - Xi|)
where = i
return where

∈ Θ(n)

6. (30P) Bir çizge (graph) iki ayrık X ve Y kümesine ayrılabiliyorsa **bipartite** çizge olarak adlandırılmaktadır, öyle ki X kümesine ait her bir düğümün tüm komşuları Y kümesine ait, benzer şekilde Y kümesine ait her bir düğümün tüm komşuları X kümesinden olmalıdır. Örneğin (i) deki çizge **bipartite** iken (ii)'deki değildir.



(i)



(ii)

Bir çizgenin bipartite olup olmadığını bulan DFS ya da BFS tabanlı bir algoritma tasarlayınız. Algoritmanın zaman karmaşıklığını belirleyiniz.

```
Count[1 ... |V|]   // Global Array
Algorithm   isBipartite (G(V,E))
                                                   print "Bipartite"
    count = 0
    cont = true
    for each vertex v in V do
        if v marked with 0
            cont = dfs(v)
            if cont == false
                print "Not bipartite"

dfs(v):

    count = count + 1
    Count[w] = count
        for each vertex w adjacent to v
            if w marked with 0
                dfs(w)

            else
                dif = count[v] - count[w]
                if dif % 2 == 0
                    return false

        return true
```