Program Studi Teknik Informatika

Institut Teknologi Sumatera

Lampung

Ujian Tengah Semester **Strategi Algoritma**

Jumat, Tanggal 11 Maret 2016

Waktu: 120 menit

Dosen: Rinaldi Munir

*Berdoalah terlebih dahulu agar Anda sukses dalam ujian ini!*

1. (*Brute Force*) Jelaskan algoritma mencari elemen terbesar di dalam larik (array) dengan algoritma *brute force.*

(Nilai: 10)

1. (*Algoritma Greedy*) Sebuah kapal besar akan diisi dengan muatan. Muatan tersebut disimpan di dalam peti kemas dan tiap peti kemas berukuran sama, tetapi berat peti kemas (yang sudah berisi muatan) berbeda belum tentu sama. Misalkan *wi* adalah berat peti kemas ke-*i*, 1 ≤ *i* ≤ *n*. Kapasitas kapal membawa muatan adalah *C*. Kita ingin memuat kapal dengan sebanyak mungkin peti kemas. Rumuskan penyelesaian persoalan ini dengan algoritma *greedy* dengan menjelaskan properti algoritma *greedy*-nya (himpunan kandidat, himpunan solusi, fungsi seleksi, fungsi kelayakan, dan fungsi obyektif apa, lalu lakukan contoh perhitungan untuk instansiasi persoalan berupa *n* = 8, *w* = (100,200,50,90,150,50,20,80), dan *C* = 400. (Nilai: 25)

1. (a) Tuliskan langkah-langkah pengurutan 7 buah elemen di bawah ini dengan algoritma *Mergesort* sehingga seluruh elemen terurut menaik. (Nilai: 10)

12, 6, 10, 25, 1, 7, 14

(b) Ulangi soal (a) di atas dengan algoritma *Quick Sort*. Elemen yang dijadikan *pivot* adalah elemen pertama:

(Nilai: 10)

1. (*Fractional Knapsack Problem*) Tinjau persoalan *Fractional* *Knapsack* dengan *n* = 6. Misalkan objek-objek tersebut kita beri nomor 1, 2, 3, 4, 5, dan 6. Properti setiap objek *i* dan kapasitas *knapsack* adalah sebagai berikut

*w*1 = 10; *p*1 = 45

*w*2 = 5; *p*2 = 10

*w*3 = 2; *p*3 = 25

*w*4 = 5; *p*4 = 30

*w*5 = 8; *p*5 = 16

*w*6 = 8; *p*6 = 12

Kapasitas *knapsack* *W* = 18

Selesaikan persoalan ini dengan algoritma *Greedy* sehingga diperoleh keuntungan yang maksimum. Solusi dinyatakan dengan tupel *X* = (*x*1, *x*2, *x*3, *x*4, *x*5, *x*6), yang dalam hal ini *xi* ∈ **R**. Hitung juga berapa jumlah keuntungan yang dapat diperoleh. (Nilai: 25)

1. (*BFS* dan *DFS*) Pohon merentang (*spanning tree*) dari sebuah graf *G* adalah pohon yang mengandung semua simpul graf. Gunakan algoritma *BFS* dan *DFS* untuk mengunjungi semua simpul di dalam graf *G* di bawah ini jika traversalnya dimulai dari simpul *e*. (Nilai: 20)

