UNIVERSITI SAINS MALAYSIA

Second Semester Examination
2015/2016 Academic Session

June 2016

CPT212 – Design & Analysis of Algorithms
[Reka Bentuk & Analisis Algoritma]

Duration : 2 hours
[Masa : 2 jam]

INSTRUCTIONS TO CANDIDATE:
[ARAHAN KEPADA CALON:]

- Please ensure that this examination paper contains **FOUR** questions in **SEVEN** printed pages before you begin the examination.

  *[Sila pastikan bahawa kertas peperiksaan ini mengandungi **EMPAT** soalan di dalam **TUJUH** muka surat yang bercetak sebelum anda memulakan peperiksaan ini.]*

- Answer **ALL** questions.

  *[Jawab **SEMUA** soalan.]*

- You may answer the questions either in English or in Bahasa Malaysia.

  [Anda dibenarkan menjawab soalan sama ada dalam bahasa Inggeris atau bahasa Malaysia.]

- In the event of any discrepancies, the English version shall be used.

  [Sekiranya terdapat sebarang percanggahan pada soalan peperiksaan, versi bahasa Inggeris hendaklah diguna pakai.]

1. (a) Two matrices of same dimension (A, B) can be added as follows:

Dua matriks dengan dimensi yang sama (A, B) boleh ditambah seperti yang berikut:

$$\begin{bmatrix} a_{1,1} & a_{1,2} \\ a_{2,1} & a_{2,2} \end{bmatrix} + \begin{bmatrix} b_{1,1} & b_{1,2} \\ b_{2,1} & b_{2,2} \end{bmatrix} = \begin{bmatrix} a_{1,1} + b_{1,1} & a_{1,2} + b_{1,2} \\ a_{2,1} + b_{2,1} & a_{2,2} + b_{2,2} \end{bmatrix}$$

(i) Write a Java function, addMatrix(int[][] a, int[][] b), to add two integer matrices and return the resulting matrix.

Tulis suatu fungsi Java untuk menambah dua matriks, addMatrix(int[][] a, int[][] b, dan memulangkan matriks yang terhasil.

(ii) For any two (2) types of primitive operations of the code written in 1(a)(i), determine the number of primitive operations and the asymptotic complexity in big-Oh notation.

Bagi mana-mana dua (2) jenis operasi primitif bagi kod yang ditulis dalam 1(a)(i), tentukan bilangan operasi primitif dan kekompleksan asimptot dalam tatatanda O-besar.

(40/100)

(b) A program X with input data 5, 6, 7, and 8 performs 17, 20, 23, and 26 operations respectively. What is the asymptotic complexity in big-Oh notation for the program?

Sebuah program X dengan data input 5, 6, 7 dan 8 melaksanakan 17, 20, 23 dan 26 operasi masing-masing. Apakah kekompleksan asimptot dalam tatatanda O-besar bagi program tersebut?

(15/100)

(c) Describe and illustrate using a figure, the operation of radix sort on the following list of words:

Huraikan dan ilustrasikan dengan sebuah gambar rajah, operasi bagi isihan radiks bagi senarai perkataan berikut:

egg, bee, apple, bean, abc, citrus

(30/100)

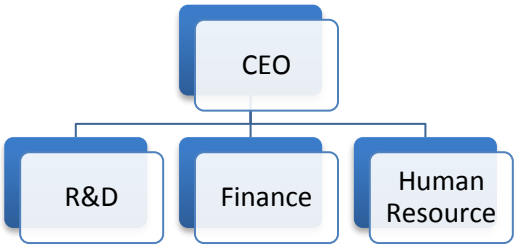
(d) What is the average asymptotic complexity of quick sort and what is the advantage of quick sort compared to merge sort?

Apakah kekompleksan asimptot purata untuk isihan cepat dan apakah kelebihan isihan cepat berbanding isihan cantum?

(15/100)

2. (a) The figure below shows an example of how a tree structure can be used to represent the information of an organization. Write a simple Java class, MultibranchTree, from scratch to implement a tree structure which will allow you to represent the information shown in the figure below.

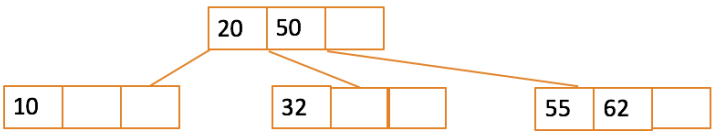
Gambar rajah di bawah menunjukkan contoh bagaimana sebuah struktur pepohon boleh digunakan untuk mewakili maklumat bagi sebuah organisasi. Tulis kelas Java yang mudah, MultibranchTree, dari mula untuk mengimplementasi sebuah struktur pepohon yang boleh mewakili maklumat seperti yang ditunjukkan dalam gambar rajah di bawah.



(25/100)

(b) Given a B-tree below. Explain what happen when the value “20” is deleted from the tree.

Diberi sebuah pepohon B di bawah. Terangkan apa akan berlaku apabila nilai “20” dihapuskan daripada pepohon tersebut.



(15/100)

(c) AVL tree can be used to keep a sorted list of words in lexicographic order.

Pepohon AVL boleh digunakan untuk menyimpan sebuah senarai perkataan terisih dalam tertib leksikografi.

(i) Assume you have the following list of words: apple, banana, papaya, lemon, lime, and watermelon. Describe and illustrate how the words are inserted into an AVL tree. Assuming the words given are used as key and value.

Andaikan anda mempunyai suatu senarai perkataan berikut: apple, banana, papaya, lemon, lime, dan watermelon. Huraikan dan ilustrasikan bagaimana perkataan-perkataan tersebut disisip ke dalam pepohon AVL. Andaikan perkataan yang diberi digunakan sebagai kekunci dan nilai.

(ii) What is the benefit of using an AVL tree, compared to a simple binary search tree? How can we print out the words in lexicographic order from an AVL tree?

Apakah kebaikan menggunakan pepohon AVL berbanding pepohon gelintaran perduaan yang mudah? Bagaimanakah kita boleh mencetak perkataan-perkataan dalam urutan leksikografi daripada pepohon AVL?

(60/100)

3. (a) A function is required to determine the number of edges in a directed graph.

Sebuah fungsi diperlukan untuk menentukan bilangan tepi dalam graf tak berarah.

(i) How can we determine the number of edges if we are given the graph in adjacency list representation? Give also the corresponding pseudocode for this purpose?

Bagaimanakah kita boleh menentukan bilangan tepi jika kita diberikan graf berkenaan dalam perwakilan senarai kesebelahan? Beri juga pseudokod berkenaan untuk tujuan ini.

(ii) What is the complexity of the algorithm in 3(a)(i). Justify your answer.

Apakah kekompleksan algoritma dalam 3(a)(i). Justifikasikan jawapan anda.

(iii) What would happen to the complexity of the algorithm in 3(a)(i) if we are given the graph in adjacency matrix representation? Justify your answer.

Apakah yang akan terjadi terhadap kekompleksan algorithma dalam 3(a)(i) jika kita diberikan graf berkenaan dalam perwakilan matriks kesebelahan. Justifikasikan jawapan anda.

(30/100)

(b) Consider the following greedy strategy for finding a shortest path from vertex *start* to vertex *goal* in a given connected graph:

1. Initialize *path* to *start.*
2. Initialize *VisitedVertices* to *{start}.*
3. If *start=goal,* return *path* and exit. Otherwise, continue.
4. Find the edge *(start,* v) of minimum weight such that v is adjacent to *start* and v is not in *VisitedVertices.*
5. Add v to *path.*
6. Add v to *VisitedVertices*
7. Set *start* equal to v and go to step 3.

Pertimbangkan strategi tamak berikut untuk mencari laluan terpendek dari bucu mula *ke bucu* matlamat *pada graf terkait yang diberi:*

1. *Awalkan* laluan *kepada* mula*.*
2. *Awalkan* BucuDilawat *kepada {mula}.*
3. *Jika* mula=matlamat*, kembalikan* laluan *dan keluar. Jika tidak, teruskan.*
4. *Cari tepi (*mula*, v) dengan pemberat minimum yang mana v adalah bersebelahan dengan* mula *dan v bukan dalam* BucuDilawat*.*
5. *Tambahkan v kepada* laluan*.*
6. *Tambahkan v kepada* BucuDilawat*.*
7. *Tetapkan* mula *kepada v dan pergi ke langkah 3.*

Does this greedy strategy always find a shortest path from *start* to *goal?* Either explain intuitively why it works, or give a counter example using the following example:

Adakah strategi tamak ini sentiasa mendapatkan laluan terpendek dari mula *ke* matlamat*? Sama ada jelaskan secara intuitif kenapa ia berjaya, atau berikan penyangkal menggunakan contoh berikut:*

A weighted graph with vertices *A*, *B*, *C* and *D*, and the following edges and corresponding weights:

Sebuah graf berpemberat dengan bucu-bucu A, B, C *dan* D*, dan tepi serta pemberat berkaitan seperti yang berikut:*

- (*A*,*B*,1)
- (*A*,*C*,2)
- (*B*,*D*,3)
- (*C*,*D*,1)

(30/100)

(c) (i) Draw the 11-entry hash table that results from using the hash function h(i) = (3i + 5) mod 11, to hash the keys 12, 44, 13, 88, 23, 94, 11, 39, 20, 16 and 5, assuming collisions are handled by chaining.

Lukis jadual cincang 11-kemasukan yang terhasil daripada penggunaan fungsi cincang h(i) = (3i + 5) mod 11, untuk mencincang kunci 12, 44, 13, 88, 23, 94, 11, 39, 20, 16 dan 5, dengan mengandaikan perlanggaran dikendalikan oleh perantaian.

(ii) What would be the result of question 3(c)(i), assuming collisions are handled by linear probing.

Apakah hasil dari soalan 3(c)(i), jika mengandaikan perlanggaran dikendalikan oleh pemeriksaan linear.

(40/100)

4. (a) (i) Given an alphabet $\{a_i\}$ with frequencies $\{f(a_i)\}$ using the Huffman algorithm, find a set of binary codewords $C = \{c(a_1), \ldots c(a_n)\}$ such that the average number of bits used to represent the data is minimized. Show the representation of the binary code as a binary tree.

Diberi satu abjad $\{a_i\}$ dengan frekuensi $\{f(a_i)\}$ menggunakan algoritma Huffman, dapatkan satu set perkataan kod perduaan $C = \{c(a_1), \ldots c(a_n)\}$ supaya purata bilangan bit yang digunakan untuk mewakili data adalah minimum. Tunjukkan perwakilan kod perduaan itu sebagai pepohon perduaan.

Character *Aksara*	a	b	c	d	e	f
Frequency *Frekuensi*	0.45	0.13	0.12	0.16	0.09	0.05

(ii) List the prefixes of the string P = "aaabbaaa" that are also suffixes of P.

Senaraikan awalan bagi rentetan P = "aaabbaaa" yang juga merupakan akhiran bagi P.

(40/100)

(b) In external fragmentation of memory management, the operating system uses a dynamic partitioning placement algorithm to decide which free block to be allocated to a process. List down all **four** (**4**) algorithms available. Which one gives the best performance?

*Dalam penyerpihan luar pengurusan ingatan, sistem pengendalian menggunakan algoritma penempatan pemetakan dinamik untuk memutuskan blok bebas yang mana akan diperuntukkan kepada sesuatu proses. Senaraikan kesemua **empat** (**4**) algoritma yang ada. Yang manakah menghasilkan prestasi terbaik?*

(20/100)

(c) Consider the following pseudocode for a simple approach to string matching:

Pertimbangkan pseudokod berikut untuk pendekatan mudah pemadanan rentetan:

```
bruteForceStringMatching(pattern P, text T)
    i = 0;
    while i ≤ |T| - |P|
        j = 0;
        while Ti == Pj and j < |P|
            i++;      // (A)
            j++;
        if j == |P|
            return match at i - |P|; // (B)
        i = i - j + 1;  // (C)
    return no match;      // (D)
```

(i) Give appropriate comments to replace (A), (B), (C) and (D) as indicated in the above pseudocode.

Beri ulasan yang sesuai bagi menggantikan (A), (B), (C) dan (D) seperti yang ditandakan dalam pseudokod di atas.

(ii) For the above algorithm, give the worst case complexity and a situation when the worst case occurs.

Bagi algoritma di atas, beri kekompleksan kes terburuk dan satu situasi apabila kes terburuk berlaku.

(iii) The above algorithm finds an occurrence of a pattern in the text and discontinues after finding the first occurence. How could the algorithm be modified so that multiple searches could be carried out?

Algoritma di atas mencari satu kejadian sebuah pola dalam teks dan berhenti apabila kejadian yang pertama ditemui. Bagaimanakah algoritma berkenaan boleh diubah suai supaya penggelintaran berganda boleh dilakukan?

(40/100)