

UNIVERSITI SAINS MALAYSIA

Peperiksaan Semester Pertama  
Sidang Akademik 1999/2000

September 1999

**CSI502 - Penyelesaian Masalah dan Pengaturcaraan**

Masa: [3 jam]

---

**ARAHAN KEPADA CALON:**

- Sila pastikan bahawa kertas peperiksaan ini mengandungi **EMPAT** soalan di dalam **SEMBILAN** muka surat yang bercetak sebelum anda memulakan peperiksaan ini.
  - Jawab **SEMUA** soalan.
  - Peperiksaan ini akan dijalankan secara 'Open Book'.
  - Anda dibenarkan membawa sebarang bahan rujukan ke bilik peperiksaan.
  - Anda boleh memilih untuk menjawab dalam Bahasa Malaysia atau Bahasa Inggeris.
-

1. (a) Dengan menggunakan perkataan anda sendiri, jelaskan mengapa kohesi yang tinggi penting dalam mereka bentuk atur cara. Jika perlu anda boleh menggunakan contoh yang bersesuaian dalam penerangan anda. [20/100]

- (b) (i) Satu algoritma diperlukan untuk membaca beberapa senarai markah peperiksaan (satu nombor dibaca setiap kali), dan memaparkan nombor-nombor tersebut sebagai lulus (markah  $\geq 50$ ) atau gagal (markah  $< 50$ ). Senarai markah ini di akhiri dengan nilai negatif. Tuliskan jawapan anda dalam bentuk carta aliran. [15/100]

- (ii) Jika sekiranya sistem mata gred (GP) baru diguna pakai dengan syarat-syarat pemetaan seperti berikut:

Markah	GP
80 - 100	4.00
70 - 79	3.67
64 - 69	3.33
58 - 63	3.00
52 - 57	2.67
46 - 51	2.33
40 - 45	2.00
36 - 39	1.67
32 - 35	1.33
28 - 31	1.00
25 - 27	0.67
0 - 24	0.00

Lakukan perubahan kepada algoritma anda di atas supaya keperluan baru ini diambil kira. Bagi setiap nilai markah yang dimasukkan oleh pengguna, sistem ini perlu melakukan perkara-perkara berikut:

- menokok nilai markah dalam setiap julat yang bersesuaian
- menambah nilai pembilang dalam setiap julat yang bersesuaian

Apabila pengguna memasukkan nilai negatif, proses pembacaan dihentikan, dan sistem ini memaparkan output berikut:

- markah purata dalam setiap julat yang bersesuaian,
- jumlah keseluruhan pelajar dalam setiap julat yang bersesuaian, dan
- nilai GP untuk setiap julat yang bersesuaian

Anda perlu menjawab soalan ini dalam bentuk pseudokod.

[25/100]

- (c) Atur cara C mengandungi beberapa pengisytiharan dan nilai awal seperti berikut:

```
int i = 13, j = 9;
char c = 'c', d = 'd';
float x = 0.5, y = -10;
```

Tentukan output setiap pernyataan C berikut. Guna nilai-nilai awal yang telah diumpukkan kepada pemboleh ubah untuk setiap pernyataan C.

- (i) `printf("%d", 3 * i - 2 * j % 2 * d - c);`
- (ii) `printf("%d", 2 * ((i/5) + (4 * (j - 3)) % (i + j - 2)));`
- (iii) `printf("%d", (j > 9) ? i : j ? j : 0);`
- (iv) `printf("%c", (c < d) ? c + 2 : d);`
- (v) `printf("%d", x + y);`

[25/100]

- (d) Dengan menggunakan contoh yang bersesuaian, tunjukkan penggunaan prapemproses `#define`. Terangkan kenapa anda perlu menggunakannya dalam atur cara C.

[15/100]

2. (a) Kaji kod berikut dengan teliti. Apakah yang akan dipaparkan oleh kod berkenaan apabila dilarikan?

```
int i, j, x=0;

for(i=4; i >= 0; i--)
    for(j=0; j < i; ++j) {
        x += i + j - 1;
        printf("%d ", x);
    }
printf("\nFinal value of x is %d\n", x);
```

[20/100]

- (b) (i) Tulis semula ungkapan pilihan berikut dalam bentuk pernyataan **if-else**.

$$\text{result} = (a > b) ? 1 : (a < b) ? -1 : 0$$

- (ii) Adakah ungkapan pilihan berikut akan menghasilkan keputusan yang sama dengan ungkapan pilihan dalam 2 (b)(i) di atas? Jelaskan jawapan anda.

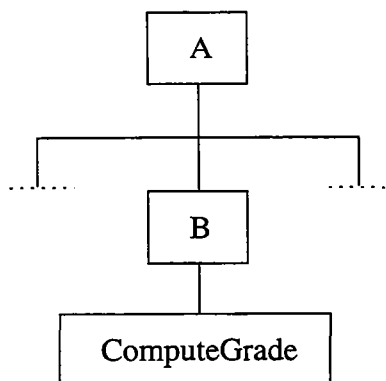
$$\text{result} = (a < b) ? -1 : (a > b)$$

[20/100]

- (c) Anda ditugaskan untuk menguji fungsi ComputeGrade yang mengira gred peperiksaan (A-D) bersesuaian dengan markah yang diberikan (dalam julat 0 hingga 100). Markah lebih besar atau sama dengan 70 akan diberikan gred A, markah dari 50 hingga 69 akan diberikan gred B, markah dari 30 hingga 49 akan diberikan gred C dan markah lebih kecil atau sama dengan 30 akan diberikan gred D. Spesifikasi fungsi ini adalah seperti berikut:

Nama:	ComputeGrade	
Argumen-argumen:	IN	Mark Markah mentah (integer)
	OUT	Grade Gred yang bersesuaian (aksara)
	OUT	OK Benar jika markah adalah sah (integer)
Penerangan:	Jika Mark adalah sah, OK akan mempunyai nilai benar dan Grade akan mengandungi nilai gred yang bersesuaian dengan Mark. Jika markah tidak sah, OK akan mengandungi nilai palsu dan nilai yang terdapat pada Grade tidak diketahui.	
Modul Pemanggil:	B	

Carta struktur berikut menunjukkan reka bentuk atur cara keseluruhan:



- (i) Katakan kaedah ujian bawah-atas dipilih, tuliskan 'driver' untuk menguji fungsi ComputeGrade, iaitu modul B.
- (ii) Kenal pasti kes-kes ujian untuk menguji fungsi ComputeGrade ini. Sediakan satu pelan ujian dalam bentuk jadual yang mengandungi data seperti data ujian, tujuan (kes-kes ujian) dan keputusan yang dijangkakan.

[35/100]

- (d) Atur cara berikut memanggil fungsi yang dinamai **mystery**:

```
#include <stdio.h>

/* function prototype */
...?...

main()
{
  int num;

  printf("Enter a number : ");
  scanf("%d",&num);

  /* function call */
  ...?...

  printf("num is %d\n",num);
}

int mystery(int n)
{
  int s, x = 0;

  for(s=1; s <= n; s++)
    x += s * s;

  return s;
}
```

Jawab soalan-soalan berikut:

- (i) Tulis prototaip fungsi **mystery** dalam atur cara di atas.
- (ii) Tulis pernyataan yang memanggil fungsi **mystery** dalam atur cara di atas.
- (iii) Jika nilai yang dimasukkan pada num adalah 3 menerusi pernyataan **scanf()**, apakah yang dipaparkan oleh atur cara ini?
- (iv) Cadangkan nama sesuai (lebih tepat) yang patut diberikan kepada fungsi **mystery** untuk menerangkan tugas yang dilakukan olehnya.

[25/100]

3. (a) Perhatikan kod fungsi berikut:

```
int func(int n)
{
  if (n == 0)
    return 0;
  else
    return 1 + func(n/10);
}
```

- (i) Pada pendapat anda, apa agaknya yang cuba dilakukan oleh fungsi **func**?
- (ii) Tulis semula fungsi rekursif **func** dalam bentuk lelaran (iteratif).

[20/100]

- (b) (i) Isytiharkan tatasusunan dua matra yang dipanggil **m**, dan awalkan elemen-elemennya dengan nilai-nilai berikut:

```
2 3 0 0
0 0 0 0
4 5 0 0
0 0 6 7
```

- (ii) Tuliskan satu fungsi yang apabila dipanggil dengan menggunakan pernyataan **transpose(m,n)**, akan menukargantikan kandungan **n** baris dan lajur pertama tatasusunan dua matra **m**. Contohnya, jika nilai **n = 3**, dan

kandungan awal  $m = \begin{bmatrix} 2 & 3 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 4 & 5 & 0 & 0 \\ 0 & 0 & 6 & 7 \end{bmatrix}$  maka akhirnya  $m = \begin{bmatrix} 2 & 0 & 4 & 0 \\ 3 & 0 & 5 & 0 \\ 0 & 0 & 0 & 6 \\ 0 & 0 & 0 & 7 \end{bmatrix}$

[25/100]

- (c) Fungsi berikut menggunakan kaedah gelintaran linear untuk mencari nilai **key** yang terkandung dalam elemen-elemen tatasusunan **x**. Jika persamaan nilai ditemui, maka indeks kepada elemen yang mana persamaan berlaku akan dipulangkan, dan jika tiada penemuan nilai -1 akan dipulangkan.

```
int linearSearch(int x[], int key, int size)
{
    int i;

    for(i=0; i < size; i++)
        if (key == x[i])
            return i;
    return -1;
}
```

- (i) Jelaskan mengapa kaedah carian linear ini kurang efisien bagi satu tatasusunan yang bersaiz besar?
- (ii) Lakukan penukaran kepada fungsi di atas supaya fungsi berkenaan memulangkan indeks elemen tatasusunan **x** yang mana penemuan terakhir dikenal pasti dan pulangkan -1 jika tiada penemuan.

[25/100]

(d) Baca dan fahami spesifikasi atur cara di bawah:

Anda dikehendaki menulis atur cara C yang lengkap untuk melakukan tugas berikut:

- Panggil fungsi yang dinamakan **GetHours()**. Fungsi ini membenarkan anda untuk memasukkan satu nilai yang akan diumpukkan kepada pemboleh ubah setempat yang dipanggil **H**. Nilai **H** kemudiannya akan dipulangkan dan diumpukkan kepada pemboleh ubah **Hours** dalam fungsi **main()**. Nilai **H** mestilah disahkan dahulu sebelum pemprosesan selanjutnya dilakukan.
- Panggil fungsi yang dinamakan **GetRate()**. Fungsi ini membenarkan anda untuk memasukkan satu nilai kepada pemboleh ubah setempat yang dipanggil **R**. Nilai **R** kemudiannya dipulangkan (setelah disahkan) dan diumpukkan kepada pemboleh ubah **Rate** dalam fungsi **main()**.
- Panggil fungsi yang dinamakan **CalcWage()** dengan menghantar nilai **Hour** dan **Rate**. Fungsi ini perlu menggunakan dua parameter formal yang dinamakan **Hr** dan **Rt** untuk mengira gaji yang perolehi menggunakan formula  $Hr * Rt$ . Hasil kiraan akan dipulangkan dan diumpukkan kepada pemboleh ubah **Wage** yang didefinasikan dalam **main()**.
- Panggil fungsi yang dinamakan **CalcScale()** dengan menghantar nilai **Wage**. Fungsi **CalcScale()** perlu menggunakan parameter formal **Wg** untuk menyimpan nilai **Wage**. Kemudiannya, fungsi ini menentukan dan memulangkan aksara skel tax berdasarkan kepada maklumat yang terdapat dalam jadual berikut:

Wg	Aksara Skala Cukai
above 3000	'A'
1500 - 2999	'B'
below 1500	'C'

Rujuk kerangka atur cara di bawah untuk membantu anda menjawab soalan ini:

```
main()
{
    int Hours;
    float Rate, Wage;
    char TaxScale;

    ... = GetHours();
    ... = GetRate();
    ... = CalcWage(..., ...);
    TaxScale = CalcScale(...);
    printf("Wage earned is %...?... ", ...);
    printf("Tax Scale is %...?... ", ...);
}
```

[30/100]

4. (a) Berdasarkan penerangan berikut, definisikan struktur data buku dan peminjam dengan menggunakan kata kunci **struct**.

"Sesebuah perpustakaan memelihara dua entiti penting iaitu buku dan peminjam. Rekod buku mengandungi data-data seperti penulis, penerbit, nombor panggilan, tahun penerbitan, edisi dan juga statusnya (ditempah, dipinjam, boleh dipinjam). Untuk rekod peminjam pula data-data berikut dikenalpasti, nama peminjam, matrik nombor peminjam, pusat pengajian, nombor telefon. Di samping itu peminjam dipecahkan kepada dua kumpulan, staf dan pelajar. Bagi kumpulan staf, ahli-ahlinya terdiri dari staf akademik dan staf bukan akademik. Begitu juga dengan kumpulan pelajar, ahli-ahlinya dikategorikan kepada pelajar sarjana dan pelajar sarjana muda. Untuk setiap kategori peminjam, jangka masa pinjaman perlu ditetapkan"

[25/100]

- (b) Diberi definisi struktur-struktur berikut:

```
struct forwarder {
    int forwarder_id;
    char address[30];
    char contact_person[20];
};

struct shipment {
    int agent_id;
    int good_code;
    char destination[50];
    struct forwarder fd;
};
```

- (i) Isytiharkan satu pemboleh ubah biasa yang dinamakan **agentA** yang jenis datanya adalah **struct shipment**, dan satu pemboleh ubah penuding yang dinamakan **agentPtr** yang jenis datanya adalah **struct forwarder**.
- (ii) Gunakan fungsi **malloc** untuk menjana ruang kepada struktur **forwarder**, dan umpukkan hasilnya iaitu alamat asas blok ingatan yang dijanakan kepada pemboleh ubah **agentPtr**.
- (iii) Umpukkan nilai-nilai berikut kepada ahli struktur **forwarder** yang dirujuk oleh pemboleh ubah **agentPtr**.

```
forwarder_id <-- 12345
address <-- "Port Klang (West)"
contact_person <-- "Mr. Bean"
```

[25/100]



- (c) Fail **RAIN.DAT** menyimpan data taburan hujan sehingga 20 tahun di Semenanjung Malaysia seperti berikut:

```

12 8 14 28 43 .... 23 10 {Jan, Feb, Mar, etc data untuk tahun 1}
6 10 0 12 67 .... 6 2 {Data untuk tahun 2}
..
..
8 7 12 7 23 .... 9 11 {Data untuk tahun n}

```

Tulis atur cara C yang akan mengira untuk setiap bulan, tahun yang mana taburan hujan adalah maksimum, dan atur cara tersebut kemudiannya akan memaparkan output berikut:

Bulan	Tahun
1	3
2	7
3	10
.	.
.	.
12	15

Jadual di atas menggambarkan untuk bulan Januari, taburan hujan tertinggi dicatatkan pada tahun 3, untuk bulan Februari, taburan hujan maksimum adalah tahun 7, dan sebagainya.

[30/100]

- (d) Terangkan apakah yang dimaksudkan dengan pewarisan dalam paradigma berorientasi objek. Guna contoh yang bersesuaian dalam penerangan anda.

[20/100]