UNIVERSITI SAINS MALAYSIA

Peperiksaan Semester Pertama
Sidang Akademik 2004/2005

Oktober 2004

CPT103/CPM211 – Struktur Data & Paradigma Pengaturcaraan

Masa : 2 jam

ARAHAN KEPADA CALON:

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

- Jawab **SEMUA** soalan.

1. (a) Nyatakan sama ada kenyataan berikut **Benar** atau **Palsu**:

(i) Satu kelas boleh mempunyai satu atau lebih pemusnah (destructor).

(ii) Pembina salinan (copy constructor) digunakan untuk menyalin satu objek yang telah wujud kepada satu objek yang baru, manakala pengendali umpukan (assignment operator) digunakan untuk menyalin satu objek yang telah wujud kepada satu objek lain yang juga telah wujud.

(iii) Dalam bahasa C++, semua pengendali boleh disaratkan (overload).

(iv) Templat memberi kemudahan dari segi guna semula perisian.

(v) Tertib sisipan (inorder) bagi pepohon perduaan (binary tree) selalu memberikan output dalam susunan menaik.

(5 markah)

(b) Senaraikan empat (4) fasa dalam kitar hayat perisian.

(4 markah)

(c) Apakah output yang akan diberikan oleh atur cara berikut:

```
ostream_iterator<int.screen(cout, " ");
vector<int> intVector;
vector<int>::iterator vecIT;

intVector.push_back(15);
intVector.push_back(2);
intVector.push_back(10);
intVector.push_back(7);
vecIt = intVector.begin();
vecIt++;
intVector.erase(vecIt);
intVector.pop_back();

copy(intVector.begin(), intVector.end(), screen);
```

(5 markah)

(d) Terangkan fungsi rakan (friend function).

(2 markah)

(e) Apakah output yang akan diberikan oleh atur cara berikut?

```
#include <iostream>
#include <string>

using namespace std;

clas baseClass
{
public:
    void print() const;

    baseclass(string s= " ", int a=0);

protected:
    int x;

private:
    string str;
};

class derivedClass: public baseClass
{
public:
    void print() const;

    derivedClass(string s = "", int a=0, int b=0);
        //Post: str=s; x=a; y=b

private:
    int y;
}

int main()
{
    baseClass baseObject("This is base class", 2);
    derivedClass derivedObject("DDDDD", 3, 7);

    baseObject.print();
    derivedObject.print();

    return 0;
}
```

(4 markah)

2. (a) (i) `orderedLinkListType` adalah satu kelas yang mempunyai senarai yang disusun mengikut turutan. Tulis satu fungsi `insertnode` untuk memasukkan nod baru bagi kelas tersebut.

(6 markah)

(ii) Tulis semula fungsi `insertnode` di atas dan semak sama ada nod yang dimasukkan itu wujud atau tidak. Sekiranya nod tersebut telah wujud, maka berikan mesej yang sesuai.

(6 markah)

(b) Rujuk pada pengisytiharan berikut:

```
class mystery
{
.
.
.
};
```

(i) Tulis pengisytiharan bagi menyarat (overload) operator >> dalam `class mystery` di atas.

(2 markah)

(ii) Tulis pengisytiharan bagi menyarat (overload) operator ++ (post increment) dalam `class mystery` di atas.

(2 markah)

(iii) Tulis definisi bagi fungsi operator ++ (post increment) di atas.

(4 markah)

3. (a) Tulis satu templat fungsi, `reverseStack` yang mengambil objek tindanan (stack) dan objek baris gilir (queue) sebagai parameter yang mengandungi jenis unsur yang sama. Fungsi `reverseStack` menggunakan baris gilir bagi menterbalikkan (reverse) susunan unsur dalam tindanan.

(5 markah)

(b) Tulis satu templat fungsi, `reverseQueue` yang mengambil objek tindanan (stack) dan objek baris gilir (queue) sebagai parameter yang mempunyai jenis yang sama. Fungsi `reverseQueue` menggunakan tindanan bagi menterbalikkan (reverse) susunan unsur dalam baris gilir.

(5 markah)

(c) Apakah output yang akan diberikan oleh atur cara berikut:

```
#include <iostream>
#include <stack>

using namespace std;

void mystery(stack<int> s, stack<int>& t);

int main()
{
    int list[] = {15, 10, 15, 20, 25};
    stack<int> s1, s2;
    for(int i = 0; i < 5; i++)
         sl.push(list[i];
    mystery(s1, s2);
    while(!s2.empty())
    {
         cout<<s2.top()<<" ";
         s2.pop();
    }
    cout<<endl;
    return 0;
}
void mystery(stack<int> s, stack<int>& t)
{
    while(!s.empty())
    {
         t.push(2 * s.top());
         s.pop();
    }
}
```

(5 markah)

(d) Senarai di bawah ialah dua susunan yang berlainan bagi nod-nod untuk satu pepohon perduaan (binary tree):

```
preorder: GFEDCBA
inorder: DEFCGBA
```

Lukis pepohon perduaan tersebut.

(5 markah)

4. (a) Tulis satu fungsi rekursi bagi mengira jumlah dalam satu senarai.

Contoh: `jumlah_senarai(Jumlah,Senarai)`

Gunakan bahasa pengaturcaraan berikut:

(i) Lisp

(5 markah)

(ii) Prolog

(5 markah)

(b) Tulis satu fungsi rekursi bagi mencari ahli yang terkecil dalam satu senarai.

Contoh: `no_terkecil(Senarai,X)`

Gunakan bahasa pengaturcaraan berikut:

(i) Lisp

(5 markah)

(ii) Prolog

(5 markah)

5. (a) Nyatakan keputusan yang akan diperolehi daripada ungkapan Lisp yang berikut:

(i) `(append 'a '(d e f))`

(2 markah)

(ii) `(cons '(a b) '(c d))`

(2 markah)

(iii) `(cdar '((a b) ( c d)))`

(2 markah)

(iv) `(cadar '((a (b)) (x y)))`

(2 markah)

(b) Gunakan `car` dan `cdr` atau `first` dan `rest` untuk mengeluarkan `B` daripada ungkapan Lisp berikut:

(i) `(A B C D)`

(2 markah)

(ii) `(A G C (D (B)) E F)`

(2 markah)

(c) Rujuk kepada pangkalan data Prolog yang diberikan dan jawab soalan di bawah:

```
/* Manusia*/
manusia(gemuk,aminah,berambut_panjang,tinggi).
manusia(gemuk,hajar,berambut_panjang,cantik).
manusia(gemuk,ravi,berambut_panjang,rendah).
manusia(gemuk,ahmad,berambut_pendek,tinggi).
manusia(kurus,siti,berambut_panjang,cantik).
manusia(kurus,ali,berambut_pendek,cantik).
manusia(sederhana,john,berambut_panjang,cantik).
manusia(sederhana,jane,berambut_pendek,cantik).
/*Tamat*/
```

Beri pertanyaan untuk mencari:

(i) Semua orang yang gemuk.

(2 markah)

(ii) Semua orang yang berambut_panjang dan gemuk.

(2 markah)

(iii) Semua orang yang gemuk dan tinggi.

(2 markah)

(iv) Untuk menyemak sama ada terdapat orang yang sederhana dan rendah.

(2 markah)

- oooOooo -