TRƯỜNG ĐẠI HỌC BÁCH KHOA TP.HCM
Khoa Khoa học & Kỹ thuật Máy tính

Sinh viên: ……
MSSV:……

Đề thi môn Kỹ Thuật Lập Trình, niên khóa 2011-2012

Thời gian : 100 phút

Tài liệu mở

Thang điểm tối đa là 10.

Phần câu hỏi (1 điểm)

Sinh viên cần chọn câu trả lời đúng nhất cho mỗi câu hỏi sau, và sinh viên chỉ cần trả lời đúng 2/3 câu hỏi.

Câu 1.
Khi cho thực thi đoạn chương trình sau:

```
1:    #include <iostream>
2:    using namespace std;
3:    class base{
4:          private:    int a;
5:          protected:  int b;
6:          public:  int c;
7:               base(){a=1; b=2; c=3;}
8:    };
9:    class deri: public base{
10:         public:  deri(){ cout << a << b << c;}
11:   };
12:   void main(){
13:         deri d;
14:   }
```

Kết quả in ra màn hình là:
A) Lần lượt hiển thị các giá trị của a, b và c
B) Báo lỗi vì a,b và c không thể truy xuất được
C) Báo lỗi vì a và b không thể truy xuất được
D) Báo lỗi vì a không thể truy xuất được
E) Tất cả các đáp án trên đều sai.

Câu 2.
Khi cho thực thi đoạn chương trình sau:

```
1:    #include <iostream>
2:    using namespace std;
3:    class MyStaticClass{
4:    public:
5:          static int value;
6:          MyStaticClass(){ value ++;}
7:          ~MyStaticClass(){}
8:    };
9:    int MyStaticClass::value;
10:   void main(){
```

```
11:         MyStaticClass::value = 0;
12:         for(int i=0; i<9; i++)
13:             MyStaticClass *c = new MyStaticClass();
14:         cout << MyStaticClass::value;
15:     }
```

Kết quả in ra màn hình là:
A) 0
B) 1
C) 9
D) Chương trình báo lỗi vì không thể truy xuất trực tiếp vào biến **value** (dòng 11 và 14)
E) Tất cả các chọn lựa trên đều sai

Câu 3.
Khi cho thực thi đoạn chương trình sau:

```
1:      #include <iostream>
2:          using namespace std;
3:      class Father{
4:          public:  Father(){cout << "F ";}
5:      };
6:      class Child: public Father{
7:          public:  Child(){ cout << "C ";}
8:      };
9:      void main(){
10:         Child c;
11:     }
```

Kết quả in ra màn hình là:
A) F
B) C
C) F C
D) C F
E) Tất cả các chọn lựa trên đều sai

Phần lập trình (9 điểm)

Một trung tâm cung cấp dịch vụ nhận tin nhắn SMS cần một phần mềm quản lý các tin nhắn. Mỗi tin nhắn gửi về trung tâm thông thường có từ hai đến ba từ, giữa hai từ luôn có một khoảng trắng. Từ đầu tiên gồm 4 ký tự số, là mã dịch vụ cần sử dụng. Từ thứ hai là tên tài khoản người sử dụng. Từ thứ ba (nếu có) là thông tin đối số. Ví dụ "**3556 huyhoang**" và "**3556 huyhoang doiso**" là các dạng tin nhắn hợp lệ, với 3556 la mã dịch vụ và huyhoang là

tên tài khoản người sử dụng. Dưới đây chúng ta sẽ cung cấp các tính năng dùng để hỗ trợ tính toán thống kê.

Để hiện thực hệ thống quản lý này, các cấu trúc, hàm và biến cho sẵn như sau :

- Cấu trúc lưu trữ thời gian:

```
struct tm{
  int sec;              // 0 đến 59
  int min;              // 0 đến 59
  int hour;             // 0 đến 23
  int mday;             // 1 đến 31
  int mon;              // 0 đến 11, 0=tháng Giêng
  int year;             // 0=năm 2000
}
```

- Cấu trúc lưu trữ tin nhắn:

```
struct msg{
  char content[3][10]; // tin nhắn được lưu bằng nhiều chuỗi con,
                       // mỗi chuỗi con lưu một từ trong tin nhắn
  struct tm t;         // thời điểm nhận tin nhắn
};
```

Ví dụ với mẫu tin nhắn **"3556 huyhoang doiso"** :

content[0] sẽ lưu từ "3556"
content[1] sẽ lưu từ "huyhoang"
content[2] sẽ lưu từ "doiso"

Các hàm cho sẵn:

```
tm now();               // lấy thời gian hiện hành
int strlen(char *s);    // trả về độ dài của chuỗi s
int strchr(char *s, char a, int i=0);// trả về vị trí xuất hiện đầu
                                     // tiên của ký tự a trong chuỗi s
                                     // tính từ chỉ số i
int strcmp(char *d, char *s); // trả về 0 nếu chuỗi d trùng với s
                              // ngược lại thì sẽ khác 0
```

- Các biến toàn cục cho sẵn :

```
struct msg sms[10000]; // danh sách tin nhắn được lưu trữ,
                       // bắt đầu tại chỉ số 0
int nmsg;              // số lượng tin nhắn đã được lưu trữ
```

Các yêu cầu dưới đây cần được thực hiện một cách độc lập (không cần theo thứ tự của các câu). Để giải quyết từng câu hỏi, sinh viên có thể sử dụng hàm được yêu cầu trong các câu hỏi khác (ngay cả khi chưa viết code trả lời).

a) (2 điểm) Hãy viết hàm ***bool compare(struct tm t1, struct tm t2)*** so sánh hai thời điểm ***t1*** và ***t2***; hàm này trả về ***true*** nếu thời điểm ***t1*** trước hoặc trùng với ***t2***, ngoài ra ***false*** sẽ được trả về.

b) (2 điểm) Hãy hiện thực hàm có prototype là **int countSMS(char* post, tm t1, tm t2)**, dùng để xác định số lượng nhận tin nhắn có mã dịch vụ ***post*** được nhận trong từ thời điểm ***t1*** đến thời điểm ***t2***.

c) (2 điểm cho sinh viên lớp thường; 1 điểm cho sinh viên lớp KSTN) Sử dụng kỹ thuật đệ qui, hãy hiện thực hàm có prototype là **int countSMS(char* post, tm t1, tm t2, int n1, int n2)** để xác định số lượng tin nhắn có mã dịch vụ ***post*** được nhận trong từ thời điểm ***t1*** đến thời điểm ***t2*** và được lưu trữ từ chỉ số ***n1*** đến chỉ số ***n2*** của dãy ***sms***.

d) (2 điểm) Hãy hiện thực một hàm thống kê có thể trả về nhiều giá trị để cho biết cùng lúc mã dịch vụ nào nhận nhiều tin nhắn nhất từ thời điểm ***t1*** đến thời điểm ***t2*** và số lượng tin nhắn của dịch vụ đó nhận được từ thời điểm ***t1*** đến thời điểm ***t2***.

Lưu ý: Sinh viên tự đề xuất prototype cho hàm đáp ứng yêu cầu là có thể trả về nhiều giá trị khác nhau sau mỗi lần gọi hàm (có nhiều phương pháp: biến toàn cục, truyền tham khảo, con trỏ, cấu trúc đặc biệt, …). Sinh viên cần dùng tối thiểu hai trong các phương pháp trên sao cho mỗi lần gọi hàm có thể trả về nhiều giá trị khác nhau.

e) (1 điểm cho sinh viên lớp KSTN; 1 điểm thưởng cho sinh viên lớp thường) Hãy hiện thực hàm có prototype là **void getKBestServices(int k, tm t1, tm t2)**, dùng để hiển thị ***k*** mã dịch vụ nào nhận nhiều tin nhắn nhất trong khoảng thời gian ***t1*** đến ***t2***.

f) (1 điểm) Trong thiết kế chương trình hiện tại, số tin nhắn tối đa có thể lưu trữ được là 10000, chỉ có tối đa 3 từ trong 1 tin nhắn, chỉ có tối đa 10 ký tự trong 1 từ. Hãy đề xuất giải pháp để loại bỏ các giới hạn trên.

g) (1 điểm thưởng) Hãy nêu các giải pháp và phân tích ưu và nhược điểm của mỗi giải pháp để các tin nhắn vẫn được lưu trữ khi chương trình kết thúc, và khi chương trình được thực thi trở lại, các tin nhắn được sử dụng trở lại.

-HẾT-

h) (1.5 điểm) Hãy hiện thực hàm có prototype là **void stockSMS(char* content)**, dùng để lưu trữ thông tin nhận được từ một tin nhắn SMS bằng cách

b1) (1 điểm) Chỉ sử dụng cấu trúc lặp ***for*** mà không dùng cấu trúc lặp khác

b2) (1 điểm) Chỉ sử dụng cấu trúc lặp ***do..while*** mà không dùng cấu trúc lặp khác

Lưu ý: chỉ lưu tin nhắn nếu từ đầu tiên trong tin nhắn có 4 ký tự; các chuỗi cần được kết thúc bằng ký tự '**\0**'và giả sử số ký tự trong một từ của tất cả các tin nhắn đều có độ dài tối đa là **9**.