ĐỀ CƯƠNG THỰC HÀNH

Môn: Lập trình hướng đối tượng (CT176)
Số tiết: 30 tiết - Số buổi thực hành: 5 buổi, thi TH vào buổi 6
Áp dụng cho học kỳ 2, năm học 2019-2020

Chú ý: Các bài thực hành được chấm điểm ngay tại lớp. Sinh viên làm xong, giơ tay yêu cầu giáo viên đến chấm (chỉ chấm chương trình thực thi được). Bài tập của buổi nào chấm điểm trong buổi đó (không chấm bù vào các buổi sau).

1. Buổi 1:

- ***Mục đích:***

  Sinh viên tập làm quen với ngôn ngữ Java.
  Làm quen với công cụ, môi trường lập trình Java: cài đặt, soạn thảo, biên dịch, thực thi, ...
  Viết các chương trình Java đơn giản sử dụng cấu trúc điều khiển, các phép chuyển đổi kiểu, nhập xuất cơ bản, xử lý chuỗi, xử lý ngoại lệ, … trong Java.

- ***Yêu cầu:***

Mở đầu: Dùng một phần mềm nào đó để soạn thảo (Notepad++ hoặc Netbeans hoặc Eclipse), biên dịch và thực thi chương trình Java đầu tiên để nhận vào từ bàn phím sau đó hiển thị ra màn hình **Năm sinh** và **Họ tên** của sinh viên.

Bài 1: (2%) Cải tiến bài *Mở đầu* để người dùng nhập **Ngày sinh** theo định dạng "DD/MM/YYYY" và nhập **Họ tên**; sau đó hiển thị ra màn hình **Năm sinh** và **Tên** của người đó. Nên thiết kế hàm tách năm sinh và tách tên riêng:

public static int tachNamSinh(String ngaySinh)

public static String tachTen(String hoTen)

Bài 2: (1%) Viết chương trình cho người dùng nhập vào một danh sách **Họ tên** sinh viên. Sau đó hiển thị tất cả những sinh viên có **Tên** do người dùng nhập vào.

Bài 3: (1%) Viết chương trình thực hiện các công việc sau:

- Cho người dùng nhập vào một danh sách các số nguyên.
- Sau đó, cho người dùng nhập vào một số nguyên **x** bất kỳ. Hiển thị ra màn hình số lượng số **x** có trong danh sách vừa nhập.
- Sắp xếp danh sách theo thứ tự tăng dần và hiển thị danh sách đã sắp xếp ra màn hình.

Về nhà:

- Cải tiến *Bài 3* để xử lý ngoại lệ khi người dùng nhập vào không phải là số nguyên: Yêu cầu người dùng nhập lại cho đến khi nào số đó là số nguyên.
- Cải tiến *Bài 1* để xử lý trường hợp người dùng nhập ngày sinh không đúng định dạng "DD/MM/YYYY"

2. Buổi 2:

- ***Mục đích:***

  Sinh viên tập làm quen với phong cách lập trình hướng đối tượng.
  Cài đặt 1 số lớp đơn giản, viết các phương thức và hàm xây dựng của lớp.
  Sử dụng các lớp vừa định nghĩa: tạo đối tượng và khai thác đối tượng.

- ***Yêu cầu:***

Bài 1: (2%) Cài đặt lớp **Diem** (Điểm trong không gian 2 chiều) gồm:
Thuộc tính: **x**, **y** là số nguyên, phạm vi **private**
Các phương thức, phạm vi **public**, bao gồm:

+ Hàm xây dựng mặc nhiên: Diem();
+ Hàm xây dựng có 2 tham số: Diem(int, int);
+ Nhập tọa độ cho điểm từ bàn phím: void nhapDiem();
+ Hiển thị ra màn hình tọa độ điểm theo dạng (x,y): void hienThi();
+ Dời điểm đi 1 độ dời (dx, dy): void doiDiem(int dx, int dy);
+ Trả về điểm đối xứng của điểm: Diem doiXung();
+ Lấy ra giá trị hoành độ của điểm: int giaTriX();
+ Lấy ra giá trị tung độ của điểm: int giaiTriY();
+ Tính khoảng cách từ điểm đó đến gốc tọa độ: float khoangCach();
+ Tính khoảng cách từ điểm đó đến 1 điểm khác: float khoangCach(Diem d);

Viết lớp **SDDiem** có chứa hàm main() khai thác lớp vừa định nghĩa:

+ Tạo ra điểm A tọa độ (1,2). Dời điểm A đến vị trí mới cách vị trí cũ một khoảng cách dx=1 và dy=2. Hiển thị tọa độ điểm A ra màn hình.
+ Tạo ra điểm B với giá trị nhập từ bàn phím. Hiển thị tọa độ điểm B ra màn hình.
+ Tạo ra điểm C đối xứng với điểm B qua gốc tọa độ. Hiển thị tọa độ điểm C ra màn hình.
+ Hiển thị ra màn hình khoảng cách từ điểm B đến tâm O.
+ Hiển thị ra màn hình khoảng cách từ điểm A đến điểm B.

Bài 2: (2%) Cài đặt lớp **PhanSo** (Phân số) gồm:

Các thuộc tính: **tuSo** và **mauSo** có kiểu số nguyên, phạm vi **private**

Các phương thức xây dựng gồm:

+ Hàm xây dựng mặc nhiên: PhanSo();
+ Hàm xây dựng gồm nhiều đối số: PhanSo(int tu , int mau);

Các phương thức khác, phạm vi **public**, bao gồm:

+ Hàm nhập giá trị cho 1 phân số. Nếu phân số vừa nhập có mẫu số = 0 thì yêu cầu nhập lại cho đến khi nào mẫu số khác 0.
+ Hàm hiển thị phân số theo dạng tu/mau hoặc -tu/mau.
  Yêu cầu: nếu tử số =0 thì chỉ in ra số 0, nếu mẫu số = 1 thì chỉ in ra tử số.
+ Hàm nghịch đảo phân số (làm thay đổi giá trị phân số) void nghichDao();
+ Hàm tính ra phân số nghịch đảo của 1 phân số (phân số sẽ giữ nguyên nhưng hàm trả ra giá trị là phân số nghịch đảo của nó). PhanSo giaTriNghichDao();
+ Hàm tính ra giá trị thực của phân số. Chẳng hạn phân số 1/2 có giá trị là 0.5
+ Hàm so sánh lớn hơn với phân số a boolean lonHon(PhanSo a);
+ Hàm cộng phân số với 1 phân số a. Kết quả của hàm là 1 phân số.
  PhanSo cong(PhanSo a);
+ Hàm cộng phân số với 1 số nguyên. Kết quả của hàm là 1 phân số.
  PhanSo cong(int a);

Viết lớp **SDPhanSo** có chứa hàm main() sử dụng lớp PhanSo:

+ Tạo phân số **a** = 3/7, **b** = 4/9. Hiển thị giá trị của chúng ra màn hình.
+ Tạo 2 phân số **x** và **y**. Nhập giá trị cho x và y từ bàn phím.
+ Hiển thị giá trị nghịch đảo của phân số x ra màn hình (không làm thay đổi giá trị của x).
+ Tính tổng của x + y và in kết quả ra màn hình.
+ Nhập vào 1 danh sách gồm **n** phân số (n: nhập từ bàn phím).
+ Hiển thị ra màn hình phân số lớn nhất trong danh sách phân số trên.

Về nhà 1: Bổ sung lớp PhanSo và SDPhanSo:

+ Hàm trừ, nhân, chia phân số với 1 phân số a. Kết quả của hàm là 1 phân số.
+ Hàm trừ, nhân, chia phân số với 1 số nguyên. Kết quả của hàm là 1 phân số.
+ Hàm rút gọn chính phân số đó.
+ Hàm trả về phân số là phân số rút gọn của phân số đó.
+ Tính tổng danh sách **n** phân số nhập vào từ bàn phím.
+ Sắp xếp danh sách **n** phân số theo thứ tự tăng dần.

Về nhà 2: Cài đặt lớp **MyDate** gồm:
Các thuộc tính: **ngay**, **thang**, **nam** có kiểu số nguyên, phạm vi **private**
Các phương thức, phạm vi **public**, bao gồm:
+ Hàm xây dựng: mặc nhiên MyDate() và hàm có 3 tham số MyDate(int, int, int).
+ Hàm hiển thị thông tin ngày ra màn hình.
+ Hàm nhập giá trị từ bàn phím, nếu không hợp lệ thì yêu cầu nhập lại.
+ Hàm kiểm tra xem ngày có hợp lệ hay không? boolean hopLe();
Ví dụ: Ngày 31/6/2000 hay 29/2/1999 là không hợp lệ.
+ Hàm tính ra ngày hôm sau là ngày nào: MyDate ngayHomSau();
Ví dụ: Gọi hàm ngayHomSau() trên đối tượng 30/06/2007 thì kết quả là 01/07/2007
+ Hàm cộng 1 Date với số ngày **n** nào đó: MyDate congNgay(int n);
Ví dụ: ngày 15/6/2000 cộng thêm 20 ngày là ngày 05/7/2000
Viết lớp **SDDate** có chứa hàm main() kiểm tra tính đúng đắn của lớp vừa định nghĩa.

3. Buổi 3:

- ***Mục đích:***

Sinh viên tiếp tục thực hành cài đặt lớp phức tạp hơn.
Định nghĩa hàm xây dựng sao chép, sao chép sâu đối tượng, …

- ***Yêu cầu:***

Bài 1: (2%) Cài đặt lớp **DoanThang** (đoạn thẳng) gồm:
+ Thuộc tính, phạm vi **private**:
d1, **d2** là 2 điểm đầu mút của đoạn thẳng; là đối tượng thuộc lớp Diem (đã có sẵn).
+ Các hàm xây dựng:
DoanThang();
DoanThang(Diem dau, Diem cuoi);
DoanThang(int ax, int ay, int bx, int by);
+ Các phương thức, phạm vi **public**:
- Nhập tọa độ của đoạn thẳng.
- Hiển thị giá trị 2 đầu mút của đoạn thẳng.
- Tịnh tiến đoạn thẳng đi 1 độ dời (dx, dy) nào đó.
- Kiểm tra 2 đoạn thẳng có trùng nhau không.
- Tính độ dài của đoạn thẳng.

Viết lớp **SDDoanThang** có chứa hàm main() thực hiện các công việc sau:
+ Tạo 2 điểm là A(2, 1), B(5, 3). Tạo đoạn thẳng AB. Tịnh tiến AB đi đoạn (1,2). Hiển thị tọa độ mới của đoạn thẳng.
+ Tạo đoạn thẳng CD. Nhập tọa độ cho đoạn thẳng CD đó là (3,3) và (6,5).
+ Kiểm tra xem đoạn thẳng CD có trùng đoạn thẳng AB không.
+ Hiển thị ra màn hình độ dài CD.

Bài 2: (2%) Cài đặt lớp **SinhVien** (sinh viên) gồm:
Thuộc tính, phạm vi **private**:
- mã số sinh viên: kiểu String
- họ tên: kiểu String
- ngày sinh: kiểu Date
- số lượng học phần đăng ký: kiểu số nguyên
- tên các học phần đã đăng ký: mảng kiểu String
- điểm của các học phần: mảng kiểu String (điểm tính theo A, B+, B, …)

Phương thức, phạm vi **public**:
+ Các hàm xây dựng
+ Hàm nhập thông tin cơ bản của sinh viên
+ Hàm nhập điểm cho các học phần của sinh viên
+ Hàm toString(): xuất ra chuỗi là tất cả thông tin của sinh viên

+ Hàm tính ra điểm trung bình của sinh viên theo thang điểm 4
+ Hàm đăng ký thêm 1 học phần cho sinh viên

Viết lớp **SDSinhVien** có chứa hàm main() thực hiện các công việc sau:

+ Tạo sinh viên **svNam**. Nhập thông tin cơ bản cho sinh viên svNam. Đăng ký thêm cho sinh viên svNam một học phần là "LTHDT". Hiển thị tất cả thông tin của svNam.
+ Tạo 1 danh sách sinh viên. Nhập thông tin cơ bản và điểm cho danh sách sinh viên trên.
+ Hiển thị thông tin của tất cả sinh viên bị cảnh báo học vụ (điểm trung bình < 1.0).
+ Hiển thị Họ tên của sinh viên có MSSV do người dùng nhập vào.

Về nhà 1:

+ Thêm hàm xóa 1 học phần của sinh viên
+ Tìm sinh viên có điểm trung bình cao nhất lớp

Về nhà 2: Cài đặt lớp **Gach** (gạch lót nền) như sau:

Thuộc tính, phạm vi **private**:

- mã số: kiểu String
- màu: kiểu String
- số lượng viên trong 1 hộp: kiểu int
- chiều dài viên gạch (tính theo cm): kiểu int
- chiều ngang viên gạch (tính theo cm): kiểu int
- giá bán 1 hộp: kiểu long

Phương thức, phạm vi **public**:

+ Các hàm xây dựng.
+ Hàm nhập thông tin cho 1 hộp gạch.
+ Hàm hiển thị thông tin của 1 hộp gạch.
+ Hàm tính ra giá bán lẻ 1 viên gạch: float giaBanLe();
  Biết rằng: giá bán lẻ sẽ cao hơn bán nguyên hộp là 20%.
+ Hàm tính ra diện tích nền tối đa có thể lót được của hộp gạch.
+ Hàm tính ra số lượng hộp gạch ít nhất cần có khi lót 1 nền có diện tích là D*N
  int soLuongHop(int D, int N)

Quy định là lót đúng chiều gạch, không cho xoay viên gạch.

Viết lớp **SDGach** có chứa hàm main() thực hiện các công việc sau:

+ Nhập 1 danh sách gồm **n** loại gạch lót nền (với n được nhập từ bàn phím).
+ Hiển ra màn hình thông tin các loại gạch vừa nhập.
+ Hiển thị ra màn hình loại gạch có chi phí lót thấp nhất (giá tiền / đơn vị diện tích).
+ Tính ra chi phí mua gạch khi ta lót 1 diện tích có chiều ngang là 5m và chiều dài là 20m khi ta dùng từng loại gạch trong danh sách trên.

4. Buổi 4:

- ***Mục đích:***

  Sinh viên thực hành cách cài đặt thừa kế trong Java.
  Thử nghiệm cách ghi đè phương thức (method overriding), liên kết động và tính đa hình.

- ***Yêu cầu:***

Bài 1: (2%) Cài đặt lớp **DiemMau** (Điểm có màu) thừa kế từ lớp Diem (định nghĩa trong buổi 2) bổ sung thêm

Thuộc tính, phạm vi **private**: **mau** (màu sắc): kiểu String

Các phương thức, phạm vi **public**:

+ Hàm xây dựng: DiemMau(int x, int y, String mau)
+ Hàm gán giá trị màu cho điểm: void ganMau(String mau)
+ Hàm nhập, hàm hiển thị thông tin
+ …

Viết lớp **SDDiemMau** có hàm main() thực hiện các công việc sau:

+ Tạo 1 điểm màu **A** có tọa độ là (5, 10) và màu là trắng. Hiển thị thông tin ra màn hình.

+ Tạo 1 điểm màu tổng quát **B**. Nhập giá trị từ bàn phím cho điểm B. Dời điểm B đi 1 độ dời (10,8). Hiển thị tọa độ điểm B ra màn hình. Gán màu mới cho điểm B là màu "Vàng". Hiển thị thông tin mới này.

+ Tạo 1 danh sách có **n** phần tử gồm *điểm* và *điểm màu*. Khi người dùng nhập phần tử nào thì hỏi xem muốn nhập điểm (bấm số 1) hay điểm màu (bấm số 2), sau đó tạo phần tử tương ứng và cho người dùng nhập thông tin. Cuối cùng, hiển thị tất cả các phần tử trong danh sách ra màn hình.

Bài 2: (2%) Cài đặt lớp **SinhVienCNTT** (Sinh viên công nghệ thông tin) thừa kế từ lớp SinhVien (đã định nghĩa trong buổi trước) như sau:

Thuộc tính, phạm vi **private**:

- **taikhoan**: kiểu String (tài khoản sử dụng trên hệ thống ELCIT)
- **matkhau**: kiểu String (mật khẩu ELCIT)
- **email**: kiểu String (email của sinh viên)

Phương thức, phạm vi **public**:

+ Các hàm xây dựng
+ Nhập thông tin
+ Nạp đè phương thức toString() để xuất ra thông tin của 1 sinh viên CNTT
+ Hàm đổi mật khẩu: void doiMatKhau(String newpass);
+ Hàm trả về địa chỉ email của sinh viên: String getEmail();

Viết lớp **SDSVCNTT** có hàm main() thực hiện các công việc sau:

+ Tạo 1 danh sách gồm **n** sinh viên CNTT. Nhập thông tin cho danh sách đó.
+ Nhập vào 1 địa chỉ email. Tìm tài khoản ELCIT của sinh viên có địa chỉ email trên. Hiển thị kết quả học tập của sinh viên đó.

Về nhà: Một nông trại có nuôi một số các con vật như sau: bò, heo, dê.
Tất cả các con vật trên đều có những thông tin chung như: giống, màu lông, cân nặng, …
nhưng tiếng kêu của các con vật là khác nhau.
Thiết kế sơ đồ thừa kế gồm các lớp:

+ Lớp **ConVat** gồm các thông tin chung của các con vật nêu trên và phương thức *keu()*.
+ Các lớp **Bo**, **Heo**, **De** thừa kế từ lớp ConVat và nạp đè phương thức *keu()*.

Viết lớp chứa hàm main() thực hiện các công việc sau:

+ Tạo ra danh sách gồm **n** con vật của cả 3 loài vật trên. Nhập thông tin cho các con vật.
+ Hiển thị tiếng kêu của **n** con vật đó.

Giả sử nông trại bổ sung thêm 1 vật nuôi khác là Gà. Hãy thêm lớp **Ga** vào thiết kế chương trình của mình và thực hiện lại. Quan sát kết quả.

5. Buổi 5:

- ***Mục đích:***

  Sinh viên thử nghiệm thêm 1 số cách nhập xuất trong Java.
  Sinh viên thực hành lập trình giao diện đồ họa cơ bản trong Java.

- ***Yêu cầu:***

Bài 1: (2%) Sử dụng lại lớp **SinhVien** của buổi thực hành 3.

a. Thử nghiệm cách nhập và lưu 1 danh sách sinh viên vào file.

b. Viết chương trình khác đọc file này để lấy lại danh sách sinh viên. Kiểm tra thông tin xem có chính xác không.

Bài 2: (2%) Tìm hiểu cách xây dựng 1 ứng dụng Java có giao diện đồ họa dùng để giải phương trình bậc 2 ở link sau: https://viettuts.vn/java-swing/giai-phuong-trinh-bac-2

a. Thay đổi chương trình sao cho giao diện hiển thị dạng ☐x2 + ☐x + ☐ = 0 (3 ô trống để nhập 3 giá trị của a, b, c)

b. Thay đổi chương trình sao cho JoptionPane hiển thị tất cả lỗi của tham số cùng 1 lượt. Ví dụ: "Hằng số a không hợp lệ. Hằng số b không được để trống."

Về nhà 1: Viết chương trình nhập vào 1 chuỗi từ bàn phím. Sau đó ghi chuỗi này vào 1 file.

Về nhà 2: Viết chương trình đọc nội dung 1 file nhị phân bất kỳ và lưu vào 1 file có tên khác.

Về nhà 3: Mở rộng về nhà 2 của buổi 5 nhưng với giao diện đồ họa. Gợi ý: dùng lớp JFileChooser để lựa chọn file.

Về nhà 4: Xây dựng ứng dụng đồ họa bằng Java có 2 chức năng:

+ Nhập thông tin cho 1 loại gạch (bài tập về nhà Buổi 3) và ghi vào file.
+ Hiển thị thông tin loại gạch được đọc từ file.