


LẬP TRÌNH HƯỚNG ĐỐI TƯỢNG

HƯỚNG DẪN THỰC HÀNH


Thiết kế Lớp & Cài đặt

Giáo viên lý thuyết:


 ThS Lê Xuân Định

lxding@fit.hcmus.edu.vn

Giáo viên thực hành:

 ThS Lê Xuân Định

lxding@fit.hcmus.edu.vn

 ThS Trần Huy Quang

thquang@fit.hcmus.edu.vn

 ThS Trần Duy Quang

tdquang@fit.hcmus.edu.vn

 Trần Minh Khoa

tranminhkhoa1402@gmail.com

1 Mục tiêu

- Thiết kế lớp đối tượng để giải bài toán thực tế.
- Cài đặt các lớp đã thiết kế.

2 Bài tập cá nhân

Bài toán lái xe du lịch: Ta cần lái những chiếc xe chạy bằng xăng từ các điểm xuất phát khác nhau để đi tham quan các điểm du lịch trên “sa mạc” (đi đường thẳng giữa các điểm). Ở đây chỉ có 1 trạm xăng duy nhất, nên chúng ta phải tính toán sao cho xe đi tham quan được càng nhiều điểm càng tốt và không bị hết xăng giữa đường.

Bài 1. Viết lớp **Điểm** (du lịch) lưu toạ độ (x, y) và có phương thức tính khoảng cách tới 1 Điểm khác. Yêu cầu: toạ độ mỗi điểm này là **cố định**, chỉ có thể lấy thông tin toạ độ chứ không thể trực tiếp thay đổi toạ độ. Gợi ý: Tuy không trực tiếp thay đổi được các thành phần toạ độ nhưng vẫn có thể copy nguyên Điểm, VD: Điểm A(1, 2.5), B(3.3, 4); A = B; A = Điểm(5, 6); Vì thế ta vẫn có thể dùng lớp Điểm này để lưu toạ độ GPS của Xe ở Bài 3 (bên dưới).

Bài 2. Viết lớp **Thời gian** biểu diễn (giờ : phút : giây) và có phương thức cộng, trừ, cộng dồn, trừ bớt, và so sánh với một 1 Thời gian khác; có phương thức lấy ra từng thành phần giờ, phút, giây, giây tổng (= giây + phút * 60 + giờ * 3600) và cả chuỗi biểu diễn với định dạng “giờ:phút:giây”. Có thể tạo ra đối tượng Thời gian từ giây tổng hoặc từ 3 thành phần (giờ, phút, giây).

Bài 3.

a. Viết lớp **Xe** (chạy xăng) theo đặc tả sau:

- Mỗi chiếc xe có một **số xe không đổi khác nhau** để xác định xe.
- **Lượng xăng** trong bình xăng chỉ *tăng* khi **đổ xăng** và *giảm* khi xe **chạy** (không có thất thoát).
 - Khi **đổ xăng** thì người sử dụng cung cấp thông tin bao nhiêu lít cần đổ thêm, và **Xe** đảm bảo lượng xăng không vượt quá **dung tích** bình xăng (không

tràn ra ngoài), cũng như không làm vơi đi lượng xăng trước khi đổ (hút bớt xăng). Phương thức đổ xăng sẽ trả về lượng xăng thực đổ (\leq lượng xăng yêu cầu đổ). Mỗi Xe đều có phương thức cho người dùng biết dung tích bình xăng của mình.

- Mỗi lần xe **chạy** thì người sử dụng cung cấp thông tin **Điểm** đến và tốc độ (km/h) để xe cập nhật **toạ độ GPS (Điểm)** của xe và lượng xăng trong bình, thông qua **độ hao xăng** (lít/km) của xe.
 - Nếu xe đủ xăng để chạy tới Điểm đến thì sẽ chạy, cập nhật toạ độ GPS và trả về **thời gian di chuyển (Thời gian)**; nếu không đủ xăng để chạy tới Điểm đến thì sẽ không chạy mà trả về "0:0:0".
 - Người sử dụng có thể hỏi toạ độ GPS của xe bất kỳ lúc nào.
- Người sử dụng không thể biết chính xác lượng xăng còn trong bình xăng nhưng có thể hỏi thông tin về **quãng đường còn chạy được**.
- Dung tích bình xăng và độ hao xăng của mỗi chiếc xe *không thay đổi tùy tiện*, nhưng *có thể* được chỉ định (khác nhau) khi sản xuất.

b. Viết lớp **Trạm xăng** lưu **toạ độ (Điểm)** của trạm và cung cấp dịch vụ **bán xăng** cho Xe. Yêu cầu: Để mua xăng, người sử dụng phải cung cấp một chiếc Xe đang cần đổ xăng cùng với lượng xăng cần đổ và Trạm xăng chỉ bán xăng (thực hiện *đổ xăng* cho Xe đó) nếu *toạ độ GPS của Xe đó cách toạ độ của Trạm xăng này không quá 10m (= 0.01km)*.

c. Viết chương trình tạo ra 1 “khu du lịch sa mạc” với nhiều Điểm trên mặt phẳng cùng với 1 Trạm xăng. Tiếp theo, tạo ra 2 chiếc Xe xuất phát ở 2 vị trí khác nhau bất kỳ (không cần trùng với các điểm du lịch) để đi du lịch. Cho mỗi Xe chạy với tốc độ ngẫu nhiên (trong 1 khoảng xác định) trong mỗi lần tham quan một Điểm du lịch. Kết thúc khi Xe bị hết xăng (không đủ xăng để quay về Trạm xăng) hoặc đã tham quan hết các Điểm du lịch. Xuất thông báo các Điểm du lịch mà mỗi Xe đã đi qua cùng *tổng Thời gian di chuyển* của từng Xe theo định dạng “giờ:phút:giây”.