# Lập trình hướng đối tượng

## Kiểm tra quá trình

Thời gian: 90 phút

## Bài 1: Mảng Số

Hãy thiết kế một lớp có tên là $NumberArray$ cho phép thể hiện một mảng **số thực** được khai báo động. Với những yêu cầu sau:

- Hàm tạo cho phép chấp nhận 1 đối số là 1 số nguyên thể hiện kích thước của mảng, hàm này sẽ tự động cấp phát động mảng để có thể lưu trữ nhiều số mà mỗ số có giá trị là 0.
- Hàm hủy (Hủy tử) sẽ giải phóng bộ nhớ được giữ bởi mảng.
- Hàm tạo cho phép nhập 2 đối số lần lượt là 1 số nguyên thể hiện kích thước của mảng và 1 số thực thể hiện giá trị mặc định của mảng (mỗi phần tử của mảng sau khi tạo có mang giá trị này).

Ngoài ra, lớp này có phải có những hàm thành viên (public) để thực hiện những công việc sau:

- $getSize$: trả về kích thước của mảng.
- $get$: chuyền vào 1 đối số là số nguyên $i$ và trả về giá trị của phần tử thứ $i$ của mảng. Lưu ý, $i$ tính từ 0. Thông báo cho người dùng và trả về -1 nếu $i$ không hợp lệ.
- $set$: chuyền vào 2 đối số lần lượt là vị trí $i$(số nguyên) và số thực $x$. Hàm sẽ thay thế giá trị thứ $i$ của mảng thành $x$. Lưu ý, $i$ tính từ 0. Thông báo cho người dùng và không thay thế nếu $i$ không hợp lệ.
- $getArgMax$: trả về ví trí của phần tử có giá trị lớn nhất.
- $getArgMin$: trả về ví trí của phần tử có giá trị nhỏ nhất.
- $print$: Hiển thị ra màn hình số lượng phần tử và giá trị của mỗi phần tử có trong mảng.

Viết chương trình thể hiện toàn bộ chức năng của các hàm ở lớp trên.

## Bài 2: Mảng Số mở rộng

Từ lớp $NumberArray$ ở bài 1, hãy bổ sung các hàm thành viên có yêu cầu sau đây (viết trên tệp khác):

- Cấu tử sao chép
- Quá tải toán tử xuất "$<<$" theo đúng cú pháp để thay thế cho hàm $print$ ở bài 1.
- Quá tải toán tử $+=$ để cho phép một mảng cộng với một số thực $x$. Khi đó, mọi giá trị của mảng sẽ cộng thêm giá trị $x$ tương ứng.
- (Nâng cao) Hãy viết thêm để chương trình trong $main$ sau chạy đúng!

```
NumberArray n(3, 4.5);
n.set(0, 5.1);
n.set(2, -3.2);
cout<<max(n)<<endl; // Hiển thị 5.1 ra màn hình.
```

Viết chương trình thể hiện chức năng của các hàm mình vừa viết.

# Bài 3: Bộ lọc mảng

Bộ lọc mảng - $FilterArray$ là lớp **trừu tượng** cho phép chuyển đổi mảng số - $NumberArray$ (ở bài 2) thành 1 $NumberArray$ khác có cùng kích thước nhưng các phần tử của mảng được theo đổi theo cùng 1 nguyên tắc nào đó: Vì thế lớp này gồm 2 hàm chính như sau:

- $transform$: **Hàm thuần ảo thuần túy** với đối số là 1 số thực, cho phép chuyển đổi số thực này thành 1 số thực khác theo 1 nguyên tắc nào đó được định nghiã sau trong các lớp kế thừa. Hàm này trả về 1 số thực sau khi chuyển đổi.

```
double FilterArray::transform(double)
```

- $doFilter$: Hàm với đối số là 1 $NumberArray$ (biến tham chiếu) trả về 1 $NumberArray$ khác là một mảng có cùng kích thước với mảng đối số là kết quả của quá trình chuyển đổi mỗi phần tử theo nguyên tắc của hàm $transform$ ở trên.

Dựa vào lớp này, hãy kế thừa và tạo ra 3 lớp dẫn xuất có mục đích như sau (chỉ được định nghĩa lại hàm $transform$ trong lớp dẫn xuất):

- $FACopy$: hàm $doFilter$ trả về một mảng mới giống hệt mảng cũ.
- $FAInteger$: hàm $doFilter$ trả về một mảng mới mà phần tử của nó là 1 số thực có giá trị bằng **phần nguyên** (Là 5 nếu số đó là 5.432) của phần tử tương ứng ở mảng cũ.
- $FAAdd$: hàm $doFilter$ trả về một mảng mới mà phần tử của nó là 1 số thực bằng giá trị là phần tử tương ứng ở mảng cũ cộng thêm một số thực tùy ý (số này là 1 biến thuộc tính của lớp, được xác định khi sử dụng cấu tử).

Viết chương trình minh họa với yêu cầu sau:

- Cho phép người dùng nhập vào 1 mảng: nhập số lượng phần tử và các phần tử của mảng với số lượng tương ứng.
- Tạo một đối tượng thuộc lớp $NumberArray$ với các giá trị vừa nhập.
- Thiết lập 1 mảng gồm 6 bộ lọc khác nhau (ít nhất 1 bộ lọc mỗi loại).
- Sử dụng 1 vòng $for$ duyệt qua tất cả các bộ lọc trong mảng để thấy được sự khác biệt của mảng

# Bài 4: Mảng mẫu

Với lớp $NumberArray$ được thiết kế ở bài trên, hãy sửa lại một chút để trở thành 1 lớp $template$ cho phép có thể dùng cho nhiều kiểu dữ liệu khác không chỉ số thực.

Hãy thực thi chương trình trình với kiểu $string$ với **ít nhất 5 hàm bất kỳ khác nhau** của lớp này có sẵn (tính cả hàm tạo) để tạo nên 1 chương trình có ý nghĩa.

# Bài 5: Đặc trưng của mảng số

Đặc trưng của 1 mảng số thực gồm các thông tin như sau:

- Số lượng phần tử
- Giá trị lớn nhất

- Vị trí số có giá trị lớn nhất
- Giá trị nhỏ nhất
- Vị trí số có giá trị nhỏ nhất

Hãy tạo một $struct$ lưu trữ thông tin về đặc trưng của 1 mảng số như mô tả trên có tên là $ArrayFeature$.

Hãy viết 1 hàm có tên là $getFeature$ có 1 tham số là 1 $NumberArray$ (ở bài 1) để lấy những đặc trừng từ 1 mảng số đó, trả về 1 $ArrayFeature$ mang các thông tin như mô tả.

Yêu cầu:

- Thực thi chương trình cho 3 đối đượng $NumberArray$ khác nhau. Chương trình sẽ lấy thông tin cho 3 $ArrayFeature$ tương ứng sử dụng hàm $getFeature$ ở trên.
- Hãy lưu 3 $ArrayFeature$ này vào 1 file nhị phân.

In [ ]: