1.Những tích chất của hướng đối tượng ( đa hình, đóng gói…) ở bài tập trên của các em thể hiện ở chỗ nào ?

Tính kế thừa: các lớp AlphaCoronaVirus và BetaCoronaVirus kế thừa (: public CoronaVirus) (gồm thuộc tính dna và resistance và các phương thức như loadDNAInformation())

Tính đa hình: các phương thức **virtual** của lớp cha được override để tạo sự khác biệt cho 2 lớp con Alpha và Beta (Alpha phân loại theo màu và clone được 1 cá thể mỗi chu kì, Beta phân loại theo Proteịn và clone được 2 cá thể mỗi chu kì, …)

Tính đóng gói: Các dữ liệu của đối tượng cùng loại được phân vào các lớp. Các thuộc tính của được để ẩn đối với các lớp khác (**private**) hoặc ít nhất đối với lớp không phải lớp con của nó (**protected**). Các thuộc tính của đối tượng không được sửa trực tiếp mà được nhập xuất bằng các hàm public **get – set** hay **reduce**...

2.Phân biệt khác nhau std::list/std::vector nằm ở class **Patient**  
  
Nếu bạn nào đang dùng std::list thì chuyển qua làm std::vector thử và ngược lại.

|  |
| --- |
| [list] m\_virusList:  list virus trong người bệnh nhân. |

list có cấu trúc dữ liệu dạng danh sách liên kết đôi (doubly-linked list): Mỗi node chứa thông tin (con trỏ) tới node liền kề trước và sau nó. Có thể duyệt danh sách theo cả 2 chiều xuôi và ngược, nhưng buộc phải bắt đầu từ đầu hoặc cuối list. Các thuật toán thêm xóa phần tử ở vị trí bất kì giống nhau và khá dễ dàng (thay đổi link giữa 2 node)

Vector có dạng mảng cấp phát động: có thể truy nhập phẩn tử nhanh như array (sử dụng offset – constant time), có thể tăng kích cỡ (vector được cấp phát bộ nhớ lớn liên tục từ đầu), nếu cần sử dụng kích vỡ vector lớn hơn thì sẽ được reallocate vùng nhớ. Dễ dàng thêm xóa phần tử ở đầu cuối nhưng với phần tử bất kì thì khó hơn list.

3. Tại sao khi tạo nhiều file , mã mỗi file header ta cần phải định nghĩa   
  
#pragma once

Hoặc

#ifndef [ten class]

#define [ten class]

#endif /\* MJMailIntro\_hpp \*/

Các định nghĩa này gọi là header guard:

Có một số đối tượng được định nghĩa có thể được sử dụng trong nhiều header. Ví dụ **enum class PROTEIN** được định nghĩa trong “CoronaVirus.h”. Khi “AlphaCoronaVirus.h” và “BetaCoronaVirus.h” gọi #include “CoronaVirus.h”, rồi chúng lại được included trong source.cpp, xảy ra việc **enum class PROTEIN** bị gọi và định nghĩa lại nhiều lần, gây ra lỗi.

Header guard được sử dụng: #pragma once là chỉ thị việc header chỉ được định nghĩa một lần.

Tương tự

#ifndef [ten class]

#define [ten class]

#endif

Có nghĩa [ten class] chỉ được define nếu nó chưa được define trước đó (ifndef)