ĐẠI HỌC QUỐC GIA THÀNH PHỐ HỒ CHÍ MINH

TRƯỜNG ĐẠI HỌC KHOA HỌC TỰ NHIÊN

KHOA CÔNG NGHỆ THÔNG TIN

BÁO CÁO BÀI TẬP VỀ NHÀ

MÔN HỌC: PHƯƠNG PHÁP LẬP TRÌNH HƯỚNG ĐỐI TƯỢNG

QUẢN LÝ CÔNG TY

Lớp: 21_4

Giảng viên: Lê Tuấn Thu

| Nhóm sinh viên thực hiện |

Nguyễn Huy Hoàng – 21120458

Hoàng Thị Khôn – 21120485

Thành phố Hồ Chí Minh, tháng 5 năm 2023

❖ CÁC THƯ VIỆN, LỚP VÀ HÀM CẦN THIẾT ĐỂ THỰC THI CHƯƠNG TRÌNH

➢ Thư viện

- Thư viện **fstream**: sử dụng để đọc / ghi file
- Thư viện **cstring**: sử dụng kiểu dữ liệu char* và dùng hàm getline()
- Thư viện **ctime**: sử dụng để tính ngày hiện tại
- **Pragma warning(disable: 1996)**: khắc phục lỗi cảnh báo compiler khi sử dụng ctime

➢ Lớp

- **Lớp NgaySinh:**
  - Dùng để lưu trữ và quản lý thông tin về ngày sinh của nhân viên
  - Cung cấp các phương thức để nhập, xuất ngày sinh và tính tuổi của nhân viên và được gọi lại trong lớp NhanVien
  - Hàm tinhTuoi():
    - Lấy thời gian hiện tại thông qua hàm time(0). Sau đó sử dụng hàm localtime() để chuyển đổi thời gian đó thành cấu trúc 'tm' bao gồm thông tin về năm, tháng, ngày, giờ, phút, giây
    - Lấy năm hiện tại (ltm->tm_year + 1900) trừ đi năm sinh của người đó (nam). Nếu tháng hiện tại (ltm->tm_mon +1) nhỏ hơn tháng sinh của người đó (thang) hoặc nếu hai tháng bằng nhau nhưng ngày hiện tại (ltm->tm_day) nhỏ hơn ngày sinh (ngay), tuổi sẽ giảm đi một đơn vị vì người đó chưa tới sinh nhật trong năm nay.

- **Lớp NhanVien:**
  - Dùng để lưu trữ thông tin của một nhân viên, bao gồm mã nhân viên, họ và tên, ngày sinh, địa chỉ, giới tính
  - Là lớp cơ sở (lớp cha) cung cấp các phương thức để nhập, xuất, tính lương dưới dạng hàm ảo để các lớp dẫn xuất (lớp con) có thể định nghĩa lại (override), các phương thức đọc, ghi thông tin của một nhân viên
  - Là thành phần của lớp CongTy dùng để quản lý danh sách nhân viên

- **Lớp NVSanXuat:**
  - Kế thừa từ lớp NhanVien và có thêm thuộc tính số ngày làm việc

- Override lại các phương thức nhập, xuất, tính lương của lớp cha

- **Lớp NVCongNhat:**
  - Kế thừa từ lớp NhanVien và có thêm thuộc tính số sản phẩm
  - Override lại các phương thức nhập, xuất, tính lương của lớp cha

❖ CHỨC NĂNG CHÍNH CỦA TỪNG HÀM

➢ Các hàm trong lớp NhanVien:

- **Các hàm dựng, hàm hủy**

```
NhanVien():
   ▪ Khởi tạo vùng nhớ con trỏ kiểu char maNV để lưu mã nhân viên
     với kích thước tối đa 6.
   ▪ Khởi tạo vùng nhớ con trỏ kiểu char hoTen để lưu họ tên với
     kích thước tối đa 51.
   ▪ Khởi tạo vùng nhớ con trỏ kiểu char diaChi để lưu địa chỉ
     nhân viên với kích thước tối đa 101.
   ▪ Khởi tạo giới tính với giá trị mặc định 0, ứng với giới tính
     nam

NhanVien(char* _maNV, char* _hoTen, NgaySinh _ngaySinh, char*
_diaChi, bool _gioiTinh):
   ▪ Cấp phát động một mảng char[6] và sao chép giá trị từ tham số
     _maNV vào
   ▪ Cấp phát động một mảng char[51] và sao chép giá trị từ tham
     số _hoTen vào
   ▪ Truyền đối tượng NgaySinh qua tham số _ngaySinh
   ▪ Cấp phát động một mảng char[101] và sao chép giá trị từ tham
     số _diaChi vào
   ▪ Khởi tạo thuộc tính giới tính và truyền vào tham số _gioiTinh

NhanVien(const NhanVien& nv):
   ▪ Khởi tạo vùng nhớ cho maNV, hoTen, diaChi với kích thước tối
     đa lần lượt 6, 51, 101.
   ▪ Sao chép thông tin từ NhanVien nv sang con trỏ this.

~NhanVien (): Hủy các vùng nhớ maNV, hoTen, diaChi.
```

- **Hàm nhập / xuất, tính lương của nhân viên**

```
void nhap(): Nhập thông tin nhân viên từ bàn phím có mã nhân
viên,họ tên, giới tính, ngày tháng năm sinh, địa chỉ.

void xuat(): Xuất thông tin nhân viên với mã nhân viên,họ tên, giới
tính, ngày tháng năm sinh, địa chỉ.
```

`double tinhLuong()`: trả về lương của nhân viên, mặc định trả về 0. Hàm này sẽ được cài đặt lại trong các lớp kế thừa từ lớp NhanVien

- **Hàm trả về mã nhân viên, họ tên, ngày sinh của nhân viên**

`char* getMaNV()`: Hàm lấy mã nhân viên trả về char*

`char* getHoTen()`: Hàm lấy họ tên nhân viên trả về char*

`NgaySinh getNgaySinh()`: trả về ngày sinh của một nhân viên dưới dạng đối tượng NgaySinh

- **Các hàm đọc / ghi file nhân viên viên**

`void ghiNhanVien(ofstream& ofs) const`: Hàm ghi thông tin nhân viên vào file với đối số `ofs: ofstream`
  - Ghi thông tin nhân viên với maNV (mã nhân viên), họ tên (hoTen), giới tính (0-Nam, 1-Nữ), ngaySinh (ngày sinh), diaChi (địa chỉ)

BỔ SUNG ĐỌC FILE

➢ Các hàm trong lớp NVSanXuat:

- **Các hàm dựng**

`NVSanXuat()`: hàm tạo mặc định, gọi đến hàm khởi tạo không đối số của lớp cha NhanVien và thiết lập giá trị mặc định cho thuộc tính soSanPham bằng 0

`NVSanXuat(char* _maNV, char* _hoTen, NgaySinh _ngaySinh, char* _diaChi, bool _gioiTinh, double _soSanPham)`: hàm tạo có đối số, gọi đến hàm tạo có đối số của lớp cha NhanVien và thiết lập giá trị của thuộc tính soSanPham bằng giá trị được truyền vào

`NVSanXuat(const NVSanXuat& nv)`: hàm tạo sao chép, gọi đến hàm tạo sao chép của lớp cha NhanVien và sao chép giá trị của thuộc tính soSanPham từ đối tượng được truyền vào

- **Các hàm nhập / xuất, tính lương**

`void nhap() override`: nhập thông tin cho đối tượng NVSanXuat, gọi đến hàm nhap() của lớp cha NhanVien để nhập các thông tin cơ bản và

yêu cầu người dùng nhập thêm giá trị của thuộc tính soSanPham với điều kiện soSanPham phải nằm trong khoảng từ 10 đến 15

`void xuat() override`: xuất thông tin của đối tượng NVSanXuat, gọi đến hàm xuat() của lớp cha NhanVien để xuất thông tin cơ bản và xuất giá trị thuộc tính soSanPham và lương của nhân viên sử dụng hàm tinhLuong()

`double tinhLuong() override`: tính toán và trả về giá trị lương của nhân viên dựa trên số sản phẩm được sản xuất

➢ Các hàm trong lớp NVCongNhat:

- **Các hàm dựng**

NVCongNhat(): hàm tạo mặc định, gọi đến hàm khởi tạo không đối số của lớp cha NhanVien và thiết lập giá trị mặc định cho thuộc tính soNgayLam bằng 0

NVCongNhat(char* _maNV, char* _hoTen, NgaySinh _ngaySinh, char* _diaChi, bool _gioiTinh, double _soNgayLam): hàm tạo có đối số, gọi đến hàm tạo có đối số của lớp cha NhanVien và thiết lập giá trị của thuộc tính soNgayLam bằng giá trị được truyền vào

NVCongNhat(const NVCongNhat& nv): hàm tạo sao chép, gọi đến hàm tạo sao chép của lớp cha NhanVien và sao chép giá trị của thuộc tính soNgayLam từ đối tượng được truyền vào

- **Các hàm nhập / xuất, tính lương**

`void nhap() override`: nhập thông tin cho đối tượng NVCongNhat, gọi đến hàm nhap() của lớp cha NhanVien để nhập các thông tin cơ bản và yêu cầu người dùng nhập thêm giá trị của thuộc tính soNgayLam với điều kiện soNgayLam phải nằm trong khoảng từ 22 đến 26

`void xuat() override`: xuất thông tin của đối tượng NVCongNhat, gọi đến hàm xuat() của lớp cha NhanVien để xuất thông tin cơ bản và xuất giá trị thuộc tính soNgayLam và lương của nhân viên sử dụng hàm tinhLuong()

`double tinhLuong() override`: tính toán và trả về giá trị lương của nhân viên dựa trên số ngày làm

➢ Các hàm trong lớp CongTy:

- **Các hàm dựng, hàm hủy, toán tử gán =**

CongTy(int _soNV = 0): Khởi tạo mảng dsnv, nhận vào số lượng nhân viên _soNV, cấp phát một mảng động gồm _soNV con trỏ đến đối tượng NhanVien

CongTy(const CongTy& ct): Hàm tạo sao chép, nhận vào một đối tượng CongTy ct. Tạo ra một đối tượng CongTy mới với mảng động của nó được sao chép từ mảng động của ct

~CongTy(): Giải phóng bộ nhớ động đã cấp phát cho mảng con trỏ và từng đối tượng con trỏ

CongTy& operator=(const CongTy& ct): toán tử gán =, nhận vào một đối tượng CongTy ct. Gi ải phóng bộ nhớ động của đối tượng gốc (nếu có) và tạo ra một đối tượng CongTy mới, với mảng động của nó được sao chép từ mảng động của ct

- **Hàm nhập / xuất danh sách nhân viên**

void nhapDSNV(): Nhập danh sách nhân viên, bao gồm số lượng, loại nhân viên và thông tin chi tiết của từng nhân viên. Gọi phương thức nhap() của lớp con tương ứng để nhập thông tin của từng nhân viên

void xuatDSNV(): Xuất danh sách nhân viên, bao gồm thông tin của từng nhân viên được gọi từ phương thức xuat() của lớp con tương ứng

- **Hàm đọc/ ghi danh sách nhân viên**

void ghiDSNhanVienVaoFile(const char* fileName):

  - Mở tập tin fileName để ghi dữ liệu thông qua đối tượng ofstream
  - Lặp qua danh sách nhân viên, dùng dynamic_cast để kiểm tra loại nhân viên sản xuất hay công nhật, sau đó gọi hàm ghiNhanVien() đã viết ở lớp NhanVien để ghi thông tin của từng sinh viên vào tập tin, ghi thêm các thuộc tính riêng chỉ có ở từng loại nhân viên sản xuất / công nhật

BỔ SUNG ĐỌC FILE

- **Hàm tính tổng lương, lương trung bình của nhân viên, ghi nhân viên có lương nhỏ hơn lương trung bình vào danh sách**

double tinhTongLuong(): Lặp từ đầu đến cuối danh sách để lấy ra tổng lương của danh sách

double tinhLuongTrungBinh(): Lấy tổng lương bình đã trả về ở hàm tinhTongLuong() chia cho số lượng nhân viên → Lương trung bình của danh sách

void ghiDSNVLuongNhoHonTrungBinh(const char* filename):

- Dùng vòng lặp chạy từ đầu đến cuối danh sách để kiểm tra và so sánh lương của từng nhân viên với lương trung bình của danh sách
- Nếu lương của sinh viên nhỏ hơn lương trung bình của danh sách và tùy vào loại nhân viên sản xuất hay công nhật thì ghi thông tin của các nhân viên này vào file bằng cách gọi đến hàm ghiNhanVien() đã viết ở lớp NhanVien và ghi thêm các thuộc tính riêng của từng loại nhân viên

- **Hàm tìm nhân viên với các điều kiện cho trước**

  `NhanVien* timNhanVienLuongCaoNhat():`

  - Nếu số lượng nhân viên của công ty là 0 thì trả về con trỏ null (không có nhân viên để tìm).
  - Duyệt qua danh sách nhân viên và so sánh lương của từng nhân viên với lương của nhân viên có lương cao nhất hiện tại. Nếu lương của nhân viên hiện tại lớn hơn lương của nhân viên có lương cao nhất, con trỏ nhanVienLuongCaoNhat sẽ được cập nhật để trỏ đến nhân viên có lương cao nhất.
  - Cuối cùng, trả về con trỏ đến nhân viên có lương cao nhất

  `void timNhanVienTheoMa():`
  - Yêu cầu người dùng nhập vào mã nhân viên cần tìm sử dụng hàm getline().
  - Lặp qua danh sách nhân viên và so sánh mã nhân viên của từng phần tử trong danh sách với mã nhân viên được nhập vào bằng hàm strcmp().
  - Nếu hai mã nhân viên giống nhau thì gọi phương thức xuat() để xuất thông tin của nhân viên đó ra màn hình. Ngược lại thông báo không tìm thấy nhân viên có mã tương ứng

  `void timNhanVienTheoTen():`

  - Yêu cầu người dùng nhập vào họ tên nhân viên cần tìm sử dụng hàm getline().
  - Lặp qua danh sách nhân viên và so sánh họ tên nhân viên của từng phần tử trong danh sách với họ tên nhân viên được nhập vào bằng hàm strcmp().
  - Nếu họ tên của hai nhân viên giống nhau thì gọi phương thức xuat() để xuất thông tin của nhân viên đó ra màn hình. Ngược lại thông báo không tìm thấy nhân viên có họ tên tương ứng

- **Hàm thêm / xóa nhân viên**

  `void themNhanVien():`

- Yêu cầu người dùng nhập loại nhân viên (sản xuất / công nhật)
- Tạo ra một đối tượng nhân viên mới tương ứng với loại được chọn và gọi phương thức nhap() để nhập thông tin cho đối tượng nhân viên mới này
- Tạo một mảng mới lớn hơn mảng ban đầu một phần tử để chứa danh sách nhân viên mới, sao chép tất cả các phần tử từ mảng ban đầu vào mảng mới, thêm đối tượng nhân viên mới vào cuối danh sách và giải phóng bộ nhớ được cấp phát cho mảng cũ
- Sau khi thêm đối tượng nhân viên mới, ghi danh sách nhân viên mới vào tập tin "ds_NhanVien.txt" và thông báo đã thêm thành công.

`void` xoaNhanVien():

- Yêu cầu người dùng nhập mã nhân viên cần xóa và tìm vị trí của đối tượng nhân viên có mã tương ứng trong danh sách
- Nếu không tìm thấy đối tượng nhân viên nào có mã tương ứng, thông báo cho người dùng và kết thúc
- Nếu tìm thấy đối tượng nhân viên, hàm sẽ giải phóng bộ nhớ được cấp phát cho đối tượng này, di chuyển tất cả các phần tử trong danh sách phía sau đối tượng bị xóa lên một vị trí và giảm số lượng nhân viên đi một
- Ghi danh sách nhân viên mới vào tập tin "ds_NhanVien.txt" và thông báo cho người dùng đã xóa thành công

- **Hàm đếm số nhân viên sinh tháng 5**

`void` demSoNhanVienSinhThang5():

- Lặp qua từng nhân viên trong danh sách, kiểm tra xem tháng sinh của nhân viên có phải là tháng 5 hay không. Nếu đúng thì tăng biến đếm count lên 1.
- Sau khi duyệt qua hết danh sách, hàm sẽ xuất ra màn hình thông báo về số lượng nhân viên sinh tháng 5 tìm được