```
#define MAX 100

struct NhanVien
{
    char maNV[8];
    char ho[10];
    char tenDem[10];
    char ten[8];
    char diaChi[10];
    int namSinh;
    unsigned int luong;
};

typedef NhanVien Data;

struct NODE
{
    Data info;
    NODE* pNext;
};

struct LIST
{
    NODE* pHead;
    NODE* pTail;
};

NODE* GetNode(Data nv)
{
    NODE* p = new NODE;
    if (p != NULL)
    {
        p->info = nv;
        p->pNext = NULL;
    }
    return p;
}

void CreateList(LIST& l)
{
    l.pHead = l.pTail = NULL;
}

int IsEmpty(LIST l)
{
    if (l.pHead = NULL)
    {
        return 1;
    }
    return 0;
}

void InsertTail(LIST& l, Data nv)
{
    NODE* new_ele = GetNode(nv);
    if (new_ele == NULL)
    {
        cout << "Loi!";
        return;
    }
    if (l.pHead == NULL)
    {
        l.pHead = new_ele;
        l.pTail = l.pHead;
        return;
    }
    l.pTail->pNext = new_ele;
    l.pTail = new_ele;
}

void InsertHead(LIST& l, Data nv)
{
    NODE* new_ele = GetNode(nv);
    if (new_ele == NULL)
    {
        cout << "Loi!";
        return;
    }
    if (l.pHead == NULL)
    {
        l.pHead = new_ele;
        l.pTail = l.pHead;
        return;
    }
    new_ele->pNext = l.pHead;
    l.pHead = new_ele;
}

void Nhap1NV(Data& nv)
{
    cout << "\nNhap ma nhan vien: ";
    cin >> nv.maNV;
    cout << endl << "Nhap Ho: ";
    cin >> nv.ho;
    cout << endl << "Nhap ten dem: ";
    cin >> nv.tenDem;
    cout << endl << "Nhap ten: ";
    cin >> nv.ten;
```

```cpp
    cout << endl << "Nhap dia chi: ";
    cin >> nv.diaChi;
    cout << endl << "Nhap nam sinh: ";
    cin >> nv.namSinh;
    cout << endl << "Nhap luong: ";
    cin >> nv.luong;
}

void NhapThuCong(LIST& l)
{
    Data nv;
    int soNV;
    CreateList(l);
    cout << "\nNhap so nhan vien can them thong tin: ";
    cin >> soNV;
    for (int i = 0; i < soNV; i++)
    {
        Nhap1NV(nv);
        InsertTail(l, nv);
    }
}

void XuatTieuDe()
{
    int i;
    cout << endl;
    //cout << "|";
    for (i = 0; i < 70; i++)
    {
        cout << '=';

    }
    cout << "\n";
    cout << setiosflags(ios::left);
    cout << "|";
    cout << setw(8) << "Ma SV"
        << "|"
        << setw(8) << "Ho"
        << "|"
        << setw(10) << "Ten lot"
        << "|"
        << setw(8) << "Ten"
        << "|"
        << setw(12) << "Dia chi"
        << "|"
        << setw(6) << "Nam Sinh"
        << "|"
        << setw(8) << "Luong"
        << "|";
    cout << endl;
    //cout << "|";
    for (i = 0; i < 70; i++)
    {
        cout << '=';
    }
}
void Xuat1NV(Data nv)
{
    cout << "\|";
    cout << setiosflags(ios::left)
        << setw(8) << nv.maNV << "|"
        << setw(8) << nv.ho << "|"
        << setw(10) << nv.tenDem << "|"
        << setw(8) << nv.ten << "|"
        << setw(12) << nv.diaChi << "|"
        << setw(8) << nv.namSinh << "|"
        << setw(8) << nv.luong << "|";
}
void XuatDS(LIST l)
{
    XuatTieuDe();
    for (NODE* p = l.pHead; p != NULL; p = p->pNext)
    {
        cout << "\n";
        Xuat1NV(p->info);
    }
    cout << "\n";
    for (int i = 0; i < 70; i++)
    {
        cout << '=';
    }
}

int DocFile(char* filename, LIST& l)
{
    Data nv;
    ifstream in(filename);
    if (!in)
    {
        return 0;
    }
    CreateList(l);
    while (!in.eof())
    {
        in >> nv.maNV;
        in >> nv.ho;
        in >> nv.tenDem;
        in >> nv.ten;
        in >> nv.diaChi;
        in >> nv.namSinh;
        in >> nv.luong;
        InsertTail(l, nv);
    }
    in.close();
    return 1;
}

int DemSLChungHo(LIST& l, char* ho)
{
    NODE* p = l.pHead;
    int demHo = 0;
    while (p != NULL)
    {
        if (_strcmpi(ho, p->info.ho) == 0)
        {
            demHo++;
        }
        p = p->pNext;
    }
    return demHo;
}

int DemSoNV(LIST& l)
{
    int i = 0;
    NODE* p = l.pHead;
    while (p != NULL)
    {
        p = p->pNext;
        i++;
    }
    return i;
}

double TBLuong(LIST& l)
{
    double luongTB;
    double tongLuong = 0;
    NODE* p = l.pHead;
    while (p != NULL)
    {
        tongLuong += p->info.luong;
        p = p->pNext;
    }
    return tongLuong / DemSoNV(l);
}

```

```
int NamSinhTB(LIST& l)
{
    double namTB;
    double tongNam = 0;
    NODE* p = l.pHead;
    while (p != NULL)
    {
        tongNam += p->info.namSinh;
        p = p->pNext;
    }
    return tongNam / DemSoNV(l);
}

void DaoNguocDS(LIST l)
{
    LIST ll;
    NODE* p = l.pHead;
    if (IsEmpty(l))
    {
        cout << "\nDanh sach rong";
        return;
    }
    CreateList(ll);
    while (p != NULL)
    {
        InsertHead(ll, p->info);
        p = p->pNext;
    }
    cout << "\nDanh sach sau khi dao nguoc:";
    XuatDS(ll);
}

void HoanVi(Data& a, Data& b)
{
    Data c = a;
    a = b;
    b = c;
}

void SXNoiBot(LIST& l)
{
    NODE* p, * q;
    for (p = l.pHead; p != NULL; p = p->pNext)
    {
        for (q = p->pNext; q != NULL; q = q->pNext)
        {
            if (q->info.namSinh<p->info.namSinh)
            {
                HoanVi(q->info,p->info);
            }
        }
    }
}

void XoaNVCoMSx(LIST& l, char x[8])
{
    NODE* p = l.pHead;
    if (_strcmpi(p->info.maNV, x) == 0)
    {
        l.pHead = p->pNext;
    }
    else
    {
        while (p != NULL)
        {
            if (_strcmpi(p->pNext->info.maNV, x) == 0)
            {
                p->pNext = p->pNext->pNext;
                break;
            }
            p = p->pNext;
        }
    }
}

void ChenTrucTiep(LIST& l)
{

    NODE* a, * b, * min;
    for (a = l.pHead, a != NULL; a = a->pNext;)
    {
        min = a;
        for (b = l.pHead; b != l.pTail; b = b->pNext)
        {
            if (b->info.luong > min->info.luong)
                HoanVi(b->info, min->info);
        }
    }
}
void DoiChoTrucTiep(LIST& l)
{
    for (NODE* a = l.pHead; a != l.pTail; a = a->pNext)
        for (NODE* b = a->pNext; b != NULL; b = b->pNext)
        {
            if (a->info.luong < b->info.luong)
                HoanVi(a->info, b->info);
        }
}
```