

Nội dung tuần 05

Ứng dụng các lớp đối tượng trong thư viện chuẩn.

Hướng dẫn

Hiểu về kiểu giá trị tự định nghĩa trong lớp đối tượng

Xét ví dụ sau:

```
class A
{
private:
    int info[10];
public:
    A()
    {
        for (int i=0; i<10; ++i)
        {
            info[i] = i+10;
        }
    }

    typedef int* indexOf;
    indexOf GetInfo(const int& i)
    {
        indexOf rt = new int;
        rt = info + i;
        return rt;
    }

    void Xuat()
    {
        for (int i=0; i<10; ++i)
        {
            cout << info[i] << ", ";
        }
        cout << endl;
    }
};

void main()
{
    A a;
    A::indexOf io = a.GetInfo(7);
    cout << *io << endl;
    *io = 10;
    a.Xuat();
}
```

Chú ý dòng tô vàng là định nghĩa kiểu dữ liệu indexOf trong lớp A. Mục đích sử dụng như là tham chiếu chỉ mục đến phần tử của A. Kết quả như hình dưới.

```
17
10, 11, 12, 13, 14, 15, 16, 10, 18, 19,
Press any key to continue . . .
```

Sử dụng iterator với các container trong thư viện chuẩn

Các **container** trong thư viện chuẩn đều cung cấp giải pháp sử dụng *iterator* để duyệt các phần tử, đặc biệt với các **container** không có toán tử chỉ mục như *list* thì bắt buộc chỉ có dùng *iterator* để duyệt phần tử. Về bản chất thì *iterator* cũng chỉ là kiểu dữ liệu mới được khai báo trong lớp **container**.

```
void main()
{
    list<int> lI;
    srand(12345);
    int n = rand() % 10 + 10;
    for (int i=0; i<n; ++i)
    {
        if (rand()%2 == 0)
        {
            lI.push_back(rand()%100);
        }
        else
        {
            lI.push_front(1000 + rand()%100);
        }
    }
    for (list<int>::iterator it=lI.begin(); it != lI.end(); it++)
    {
        cout << *it << ", ";
    }
    cout << endl;
}
```

Bài Số nguyên lớn

- Phần khai báo

```
class SoNguyenLon
{
private:
    vector<unsigned char> lCS;
    void Pow10(const int& n);

public:
    SoNguyenLon(void);
    SoNguyenLon(const int& cs, const int& scs);
    SoNguyenLon(const unsigned long& n);
    SoNguyenLon(const SoNguyenLon& snl);
    ~SoNguyenLon(void);

    int SoCS();
    SoNguyenLon operator+(SoNguyenLon snl);
}
```

```
SoNguyenLon operator-(SoNguyenLon snl);
SoNguyenLon operator*(SoNguyenLon snl);
bool operator>(SoNguyenLon& snl);
const SoNguyenLon& operator=(const SoNguyenLon& snl);
SoNguyenLon& operator+=(SoNguyenLon snl);
friend SoNguyenLon operator+(const unsigned int& n, const SoNguyenLon& snl);
friend SoNguyenLon operator-(const unsigned int& n, const SoNguyenLon& snl);
friend ostream& operator<<(ostream& os, SoNguyenLon& snl);

};
```

- Phần cài đặt

```
void SoNguyenLon::Pow10(const int& n)
{
    for (int i=0; i<n; ++i)
    {
        LCS.insert(LCS.begin(), 0);
    }
}

SoNguyenLon::SoNguyenLon(void)
{
    LCS.push_back(0);
}

SoNguyenLon::SoNguyenLon(const int& cs, const int& scs)
{
    int csR = cs;
    if (csR < 1)
    {
        csR = 1;
    }
    if (csR > 9)
    {
        csR = 9;
    }
    int soCS = abs(scs);
    if (soCS < 1)
    {
        soCS = 1;
    }
    for (int i=0; i<soCS; ++i)
    {
        LCS.push_back(csR);
    }
}

SoNguyenLon::SoNguyenLon(const unsigned long& n)
{
    unsigned long temp = n;
    while (temp > 9)
    {
        LCS.push_back(temp % 10);
        temp /= 10;
    }
    LCS.push_back(temp % 10);
}
```

```
SoNguyenLon::SoNguyenLon(const SoNguyenLon& snl)
{
    lCS = snl.lCS;
}

SoNguyenLon::~SoNguyenLon(void)
{
}

int SoNguyenLon::SoCS()
{
    return lCS.size();
}

bool SoNguyenLon::operator>(SoNguyenLon& snl)
{
    if (lCS.size() > snl.lCS.size())
    {
        return true;
    }
    if (lCS.size() < snl.lCS.size())
    {
        return false;
    }
    for (int i=-1; i>lCS.size(); --i)
    {
        if (lCS[i] == snl.lCS[i])
        {
            continue;
        }
        if (lCS[i] > snl.lCS[i])
        {
            return true;
        }
        return false;
    }
    return false;
}

const SoNguyenLon& SoNguyenLon::operator=(const SoNguyenLon& snl)
{
    lCS = snl.lCS;
    return *this;
}

SoNguyenLon& SoNguyenLon::operator+=(SoNguyenLon snl)
{
    int soCSMin = (SoCS() < snl.SoCS()) ? SoCS() : snl.SoCS();
    int nho = 0, temp;
    unsigned char *pTemp;
    for (int i=0; i<soCSMin; ++i)
    {
        pTemp = &lCS[i];
        *pTemp += snl.lCS[i] + nho;
        nho = *pTemp / 10;
        *pTemp %= 10;
    }
}
```

```
if (SoCS() == soCSMin)
{
    for (int i=soCSMin; i<snl.SoCS(); ++i)
    {
        temp = snl.lCS[i] + nho;
        lCS.push_back(temp % 10);
        nho = temp / 10;
    }
}
else
{
    for (int i=soCSMin; i<SoCS(); ++i)
    {
        pTemp = &lCS[i];
        *pTemp += nho;
        nho = *pTemp / 10;
        *pTemp %= 10;
        if (nho == 0)
        {
            break;
        }
    }
}
if (nho > 0)
{
    lCS.push_back(1);
}
return *this;
}

SoNguyenLon SoNguyenLon::operator+(SoNguyenLon snl)
{
    SoNguyenLon snlKQ;
    snlKQ.lCS.clear();
    SoNguyenLon *snlSCSMax = (SoCS() > snl.SoCS()) ? this : &snl;
    SoNguyenLon *snlSCSMin = (SoCS() < snl.SoCS()) ? this : &snl;
    int nho = 0, temp;
    for (int i=0; i<snlSCSMin->SoCS(); ++i)
    {
        temp = lCS[i] + snl.lCS[i] + nho;
        snlKQ.lCS.push_back(temp % 10);
        nho = temp / 10;
    }
    for (int i=snlSCSMin->SoCS(); i<snlSCSMax->SoCS(); ++i)
    {
        temp = snlSCSMax->lCS[i] + nho;
        snlKQ.lCS.push_back(temp % 10);
        nho = temp / 10;
    }
    if (nho > 0)
    {
        snlKQ.lCS.push_back(1);
    }
    return snlKQ;
}

SoNguyenLon SoNguyenLon::operator-(SoNguyenLon snl)
```

```
{
    SoNguyenLon snlKQ;
    snlKQ.lcs.clear();
    int nho = 0, i, temp;
    if (SoCS() >= snl.SoCS())
    {
        for (i=0; i<snl.SoCS(); ++i)
        {
            temp = lcs[i] - snl.lcs[i] - nho;
            nho = 0;
            if (temp < 0)
            {
                temp += 10;
                nho = 1;
            }
            snlKQ.lcs.push_back(temp);
        }
        for (; i<SoCS(); ++i)
        {
            temp = lcs[i] - nho;
            nho = 0;
            if (temp < 0)
            {
                temp += 10;
                nho = 1;
            }
            snlKQ.lcs.push_back(temp);
        }
        while(snlKQ.lcs[snlKQ.lcs.size() - 1] == 0)
        {
            snlKQ.lcs.pop_back();
        }
    }
    return snlKQ;
}

SoNguyenLon SoNguyenLon::operator*(SoNguyenLon snl)
{
    SoNguyenLon snlKQ, *psnlTemp;
    SoNguyenLon *snlSCSMax = (SoCS() > snl.SoCS()) ? this : &snl;
    SoNguyenLon *snlSCSMin = (SoCS() < snl.SoCS()) ? this : &snl;
    int nho = 0, temp;
    for (int i=0; i<snlSCSMin->SoCS(); ++i)
    {
        psnlTemp = new SoNguyenLon;
        psnlTemp->lcs.clear();
        nho = 0;
        for (int j=0; j<snlSCSMax->SoCS(); ++j)
        {
            temp = snlSCSMin->lcs[i] * snlSCSMax->lcs[j] + nho;
            psnlTemp->lcs.push_back(temp % 10);
            nho = temp / 10;
        }
        if (nho > 0)
        {
            psnlTemp->lcs.push_back(nho);
        }
    }
}
```

```
        psnlTemp->Pow10(i);
        snlKQ += *psnlTemp;
        delete psnlTemp;
    }
    return snlKQ;
}

SoNguyenLon operator+(const unsigned int& n, const SoNguyenLon& snl)
{
    SoNguyenLon snlTemp(n);
    SoNguyenLon snlKQ = snlTemp + snl;
    return snlKQ;
}

SoNguyenLon operator-(const unsigned int& n, const SoNguyenLon& snl)
{
    SoNguyenLon snlTemp(n);
    return snlTemp - snl;
}

ostream& operator<<(ostream& os, SoNguyenLon& snl)
{
    for(int i=snl.lcs.size()-1; i>=0; --i)
        os << (int)snl.lcs[i];
    return os;
}
```

Bài tập

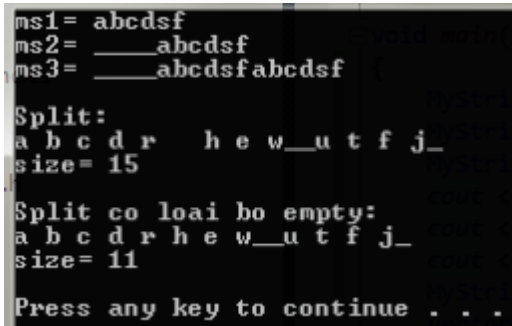
Bài 1

Xây dựng lớp MyString sao cho hàm main sau chạy đúng:

```
void main()
{
    MyString ms1("abcdf");
    MyString ms2 = "____" + ms1;
    MyString ms3 = ms1 + ms2;
    cout << "ms1= " << ms1 << endl;
    cout << "ms2= " << ms2 << endl;
    cout << "ms3= " << ms3 << endl << endl;
    MyString ms = "a,b,c;d.r;.,h;e,w__u,t.f;j_..";
    vector<char> arrChar;
    arrChar.push_back(',');
    arrChar.push_back('.');
    arrChar.push_back(';');
    vector<MyString> vMs = ms.Split(arrChar, false);
    cout << "Split:" << endl;
    for (vector<MyString>::iterator itMS=vMs.begin(); itMS!=vMs.end(); itMS++)
    {
        cout << *itMS << " ";
    }
    cout << endl << "size= " << vMs.size() << endl << endl;
    vMs = ms.Split(arrChar, true);
}
```

```
cout << "Split co loai bo empty:" << endl;
for (vector<MyString>::iterator itMS=vMs.begin(); itMS!=vMs.end(); itMS++)
{
    cout << *itMS << " ";
}
cout << endl << "size= " << vMs.size() << endl << endl;
}
```

Kết quả gọi ý



```
ms1= abcdsf
ms2= ____abcdsf
ms3= ____abcdsfabcdsf

Split:
a b c d r   h e w__u t f j_
size= 15

Split co loai bo empty:
a b c d r h e w__u t f j_
size= 11

Press any key to continue . . .
```

Bài 2

Viết chương trình đếm số lượng từ có trong 1 file văn bản tiếng Anh.