

Mục tiêu

Làm bài tập về xử lý chuỗi ký tự.

Nội dung

Sửa Bài 3 của tuần 03

```
#include <stdio.h>
#include <conio.h>
#include <stdlib.h>
#include <math.h>

#define PI 3.14156

struct Diem
{
    float x,y;
};

Diem* TaoDiem(float xV, float yV)
{
    Diem *p = (Diem*)malloc(sizeof(int));
    if (p != NULL)
    {
        p->x = xV;
        p->y = yV;
    }
    return p;
}

void PrintDiem(Diem *p)
{
    printf("(%.2f,%.2f)", p->x, p->y);
}

struct DuongTron
{
    Diem *tam;
    float banKinh;
};

DuongTron* TaoDuongTron(float xV, float yV, float rV)
{
    DuongTron *p = (DuongTron*)malloc(sizeof(DuongTron));
    if (p != NULL)
    {
        p->tam = TaoDiem(xV, yV);
        p->banKinh = abs(rV);
    }
    return p;
}

void XoaDuongTron(DuongTron *p)
```

```
{
    if (p != NULL)
    {
        free(p->tam);
        free(p);
    }
}

void PrintDuongTron(DuongTron *p)
{
    printf("[");
    PrintDiem(p->tam);
    printf(", %.2f]", p->banKinh);
}

float ChuViDuongTron(DuongTron *p)
{
    return 2*PI*p->banKinh;
}

float DienTichDuongTron(DuongTron *p)
{
    return PI * p->banKinh * p->banKinh;
}

DuongTron* ChuViLonNhat(DuongTron* *l, int n)
{
    DuongTron *p;
    float pTemp = 0, pCur;
    for (int i=0; i<n; ++i)
    {
        pCur = ChuViDuongTron(l[i]);
        if (pTemp < pCur)
        {
            pTemp = pCur;
            p = l[i];
        }
    }
    return p;
}

float TongDienTich(DuongTron* *l, int n)
{
    float s = 0;
    for (int i=0; i<n; ++i)
    {
        s += DienTichDuongTron(l[i]);
    }
    return s;
}

int XuatCacDuongTronDTLonHonTB(DuongTron* *l, int n)
{
    float sTB = TongDienTich(l, n) / n, sCur;
    printf("S trung binh: %.2f\n", sTB);
    int dem = 0;
    for (int i=0; i<n; ++i)
```

```
{
    sCur = DienTichDuongTron(l[i]);
    if (sCur > sTB)
    {
        PrintDuongTron(l[i]);
        printf(" voi S=%0.2f\n", sCur);
        dem++;
    }
}
return dem;
}

void XuatDuongTronChuViGiam(DuongTron* *l, int n)
{
    DuongTron *pTemp;
    for (int i=0; i<n-1; ++i)
        for (int j=i+1; j<n; ++j)
        {
            if (ChuViDuongTron(l[i]) < ChuViDuongTron(l[j]))
            {
                pTemp = l[j];
                l[j] = l[i];
                l[i] = pTemp;
            }
        }

    for (int i=0; i<n; ++i)
    {
        PrintDuongTron(l[i]);
        printf(" voi P=%0.2f\n", ChuViDuongTron(l[i]));
    }
}

void XoaDsDuongTron(DuongTron* *l, int n)
{
    for (int i=0; i<n; ++i)
    {
        free(l[i]);
    }
    free(l);
}

void main()
{
    srand(4003);

    // số lượng đường tròn
    int n = 5 + rand()%20;

    // tạo mảng các đường tròn
    DuongTron* *listDT = (DuongTron**)malloc(n * sizeof(DuongTron*));
    if (listDT == NULL)
    {
        return;
    }
    float xV, yV, rV;
```

```
for (int i=0; i<n; ++i)
{
    xV = (rand()%1000)*1.0f/50;
    yV = (rand()%1000)*1.0f/50;
    rV = (rand()%1000+1)*1.0f/50;
    listDT[i] = TaoDuongTron(xV, yV, rV);
}

// Xuất đường tròn có chu vi lớn nhất
printf("Duong tron co chu vi lon nhat: ");
DuongTron *pDTPMax = ChuViLonNhat(listDT, n);
PrintDuongTron(pDTPMax);
printf("\n\n");

// Tổng diện tích các đường tròn
printf("Tong dien tich: %0.2f", TongDienTich(listDT, n));
printf("\n\n");

// Xuất các đường tròn có diện tích lớn hơn diện tích trung bình
printf("Cac duong tron co dien tich lon hon TB:\n");
int dem = XuatCacDuongTronDTLonHonTB(listDT, n);
printf("So luong: %d", dem);
printf("\n\n");

// Xuất ds đường tròn giảm dần theo chu vi
printf("Ds duong tron chu vi giam dan:\n");
XuatDuongTronChuViGiam(listDT, n);
printf("\n\n");

// thu hồi bộ nhớ
XoaDsDuongTron(listDT, n);

getch();
}
```

Bài tập

Bài 1

Xây dựng struct `Nguoi` để hàm main sau chạy đúng:

```
#define MAXLEN 100
struct Nguoi
{
    char HoTen[MAXLEN];
    char DiaChi[MAXLEN];
};

Nguoi* TaoNguoi(const char *pHt, const char *pDc)
{
}

char* XuatNguoi(Nguoi *pN)
{
}
```

```
}

void ChuanHoaTen(Nguoi *pN)
{
}

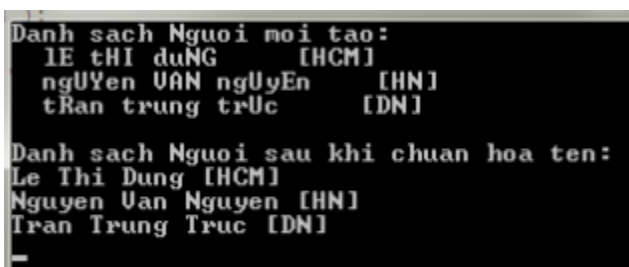
void main()
{
    int n = 3;
    Nguoi* pDsNguoi = (Nguoi**)malloc(n * sizeof(Nguoi*));
    pDsNguoi[0] = TaoNguoi(" lE tHI duNG ", "HCM");
    pDsNguoi[1] = TaoNguoi(" ngUYen VAN ngUyEn ", "HN");
    pDsNguoi[2] = TaoNguoi(" tRan trung trUc ", "DN");

    printf("Danh sach Nguoi moi tao:\n");
    for (int i=0; i<n; ++i)
    {
        printf("%s\n", XuatNguoi(pDsNguoi[i]));
    }

    printf("\nDanh sach Nguoi sau khi chuan hoa ten:\n");
    for (int i=0; i<n; ++i)
    {
        ChuanHoaTen(pDsNguoi[i]);
        printf("%s\n", XuatNguoi(pDsNguoi[i]));
    }

    for (int i=0; i<n; ++i)
    {
        free(pDsNguoi[i]);
    }
    free(pDsNguoi);
    getch();
}
```

Kết quả:



```
Danh sach Nguoi moi tao:
lE tHI duNG [HCM]
ngUYen VAN ngUyEn [HN]
tRan trung trUc [DN]

Danh sach Nguoi sau khi chuan hoa ten:
Le Thi Dung [HCM]
Nguyen Van Nguyen [HN]
Tran Trung Truc [DN]
```

Bài 2

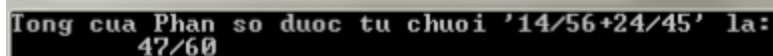
Xây dựng struct PhanSo để hàm main sau chạy đúng:

```
struct PhanSo
{
    int tu, mau;
};

char* XuatPhanSo(PhanSo *pPS)
{
```

```
}  
  
PhanSo* TongPhanSoTuChuoai(const char *s)  
{  
}  
  
void main()  
{  
    PhanSo *pPS = TongPhanSoTuChuoai("14/56+24/45");  
    printf("Tong cua Phan so duoc tu chuoai '14/56+24/45' la:\n");  
    printf("\t%s\n", XuatPhanSo(pPS));  
    free(pPS);  
  
    getch();  
}
```

Kết quả:



```
Tong cua Phan so duoc tu chuoai '14/56+24/45' la:  
47/60  
_
```

Bài 3

Viết chương trình thao tác với danh sách các tài khoản người dùng

Thông tin tài khoản:

1. Tên đăng nhập
 - a. Tên đăng nhập ko được chứa chữ “admin”
2. Mật khẩu (khi nhập thì cần nhập mật khẩu 2 lần)
 - a. Từ 6 kí tự trở lên
 - b. Có kí tự in hoa
 - c. Có kí tự $a \rightarrow z$, $A \rightarrow Z$
 - d. Có số
 - e. Có 1 trong các dấu phẩy, chấm, hỏi, hai chấm, chấm phẩy, ngã, gạch ngang, gạch dưới
3. Họ tên
 - a. Khi nhập, nhập chung 1 chuỗi họ tên, vì là họ tên nên ít nhất phải có 2 từ. Khi xuất, xuất riêng 3 phần: họ, tên lót và tên
4. Email
 - a. Phải chứa dấu @ và dấu .
5. Ngày tháng năm sinh
 - a. Khi nhập thì nhập ngày tháng năm sau, khi xuất thì xuất tuổi

Các chức năng:

1. Nhập vào danh sách tài khoản
2. Xuất danh sách tài khoản ra màn hình

❖ Hướng dẫn:

Thực hành Kỹ thuật lập trình

Tạo 5 file, viết các lệnh `#ifndef` và lệnh `#include` tương ứng

1. TaiKhoan.h
2. TaiKhoan.cpp
3. MangTaiKhoan.h
4. MangTaiKhoan.cpp
5. Main.cpp

Khai báo struct Ngày, TaiKhoan

```
// 3 dòng lệnh này để tránh include 2 lần
#ifndef _TAIKHOAN_H_
#define _TAIKHOAN_H_

// Chuyển qua hệ thống nhập xuất C++
#include<iostream>
using namespace std;
// Ko #include<stdio.h> nữa

// Có tên đăng nhập, mật khẩu, email, họ tên, ngày tháng năm sinh
// => struct Ngày
struct Ngày
{
    int ngay, thang, nam;
};
```

```
// => struct TaiKhoan
struct TaiKhoan
{
    char tenDangNhap[31]; // chuỗi tối đa 31-1=30 kí tự
    char matKhau[31];    // chuỗi tối đa 31-1=30 kí tự
    char hoTen[51];      // chuỗi tối đa 51-1=50 kí tự
    char email[51];      // chuỗi tối đa 51-1=50 kí tự
    Ngày ntns;           // ngày tháng năm sinh
};
```

Hàm nhập thông tin tài khoản

```
// Tên hàm: nhập tài khoản  
// Tham số: tài khoản chứa giá trị nhập vào  
// Trả về: ko trả về gì hết  
void Nhap(TaiKhoan &tk);
```

```
// Tên hàm: kiểm tra tên đăng nhập  
// Tham số: tên đăng nhập  
// Trả về: 0: ko hợp lệ; 1: hợp lệ -> int  
int KiemTraTenDangNhap(char tenDangNhap[]);  
  
// Tên hàm: kiểm tra mật khẩu  
// Tham số: mật khẩu  
// Trả về: 0: ko hợp lệ; 1: hợp lệ -> int  
int KiemTraMatKhau(char matKhau[]);  
  
// Tên hàm: kiểm tra họ tên  
// Tham số: họ tên  
// Trả về: 0: ko hợp lệ; 1: hợp lệ -> int  
int KiemTraHoTen(char hoTen[]);  
  
// Tên hàm: kiểm tra email  
// Tham số: email  
// Trả về: 0: ko hợp lệ; 1: hợp lệ -> int  
int KiemTraEmail(char email[]);  
  
// Tên hàm: nhập ngày tháng năm  
// Tham số: ngày tháng năm chứa giá trị nhập vào  
// Trả về: ko có  
void Nhap(Ngay &ng);
```



```
void Nhap(TaiKhoan &tk)
{
    do
    {
        //printf("Nhap ten dang nhap");
        cout << "Nhap ten dang nhap: ";
        //gets(tk.tenDangNhap);
        cin.get(tk.tenDangNhap, 30, '\n');
    } while (KiemTraTenDangNhap(tk.tenDangNhap) == 0);

    do
    {
        // printf("Nhap mat khau: ");
        cout << "Nhap mat khau: ";
        // gets(tk.matKhau);
        cin.get(tk.matKhau, 30, '\n');
    } while (KiemTraMatKhau(tk.matKhau) == 0);

    do
    {
        cout << "Nhap ho ten: ";
        cin.get(tk.hoTen, 50, '\n');
    } while (KiemTraHoTen(tk.hoTen) == 0);

    do
    {
        cout << "Nhap email: ";
        cin.get(tk.email, 50, '\n');
    } while (KiemTraEmail(tk.email) == 0);

    cout << "Nhap ngay thang nam sinh";
    Nhap(tk.nts);
}
```

```
int KiemTraTenDangNhap(char tenDangNhap[])
{
    // B1: Copy qua chuỗi s, tránh thay đổi chuỗi tenDangNhap
    char s[31];
    strcpy(s, tenDangNhap);
    // B2: Chuyển s thành chữ thường
    strlwr(s);
    // B3: Dùng hàm strstr kiểm tra chuỗi có nằm trong chuỗi
    char *p = strstr(s, "admin");
    if (p == NULL) // có
        return 0;
    return 1;
}
```

```
int KiemTraMatKhai(char matKhai[])
{
    // B1: Mật khẩu >=6 kí tự
    int doDai = strlen(matKhai);
    if (doDai < 6)
        return 0;
    // B2: Mật khẩu có kí tự in hoa
    // Duyệt qua từng kí tự [i] trong chuỗi,
    // xét xem kí tự đó có nằm trong đoạn A-Z ko
    int coKiTuHoa = 0; // giả sử chưa có kí tự in hoa
    for (int i = 0; i < doDai; i++)
    {
        if (matKhai[i] >= 'A' && matKhai[i] <= 'Z')
            coKiTuHoa = 1;
    }
    if (coKiTuHoa == 0)
        return 0;
```

```
    // B3: Mật khẩu có kí tự chữ cái
    int coKiTu = 0;
    for (int i = 0; i < doDai; i++)
    {
        if (matKhai[i] >= 'a' && matKhai[i] <= 'z')
            coKiTu = 1;
        if (matKhai[i] >= 'A' && matKhai[i] <= 'Z')
            coKiTu = 1;
    }
    if (coKiTu == 0)
        return 0;

    // B4: Mật khẩu có kí tự số
    int coSo = 0;
    for (int i = 0; i < doDai; i++)
    {
        if (matKhai[i] >= '0' && matKhai[i] <= '9')
            coKiTu = 1;
    }
    if (coSo == 0)
        return 0;
```

```
// B5: Mật khẩu có các dấu yêu cầu
int coDau = 0;
for (int i = 0; i < doDai; i++)
{
    if (matKhau[i] == ',' || matKhau[i] == '.' || matKhau[i] == ':')
        coKiTu = 1;
}
if (coSo == 0)
    return 0;
// B6: Lọt đến đây là thỏa hết các yêu cầu
return 1;
}
```

```
int KiemTraHoTen(char hoTen[])
{
    // B1: họ tên có 2 từ -> có khoảng trắng
    // Duyệt qua từng kí tự [i] trong chuỗi
    int coKhoangTrang = 0; // giả sử chưa có khoảng trắng
    int doDai = strlen(hoTen);
    for (int i = 0; i < doDai; i++)
    {
        if (hoTen[i] == ' ')
            coKhoangTrang = 1;
    }
    if (coKhoangTrang == 0)
        return 0;
    return 1;
}
```

```
int KiemTraEmail(char email[])
{
    // B1: Kiểm tra có @ ko, nếu ko return 0
    // B2: Kiểm tra có . ko, nếu ko return 0
    return 1;
}
```