TRƯỜNG ĐẠI HỌC KHOA HỌC TỰ NHIÊN TP.HCM
KHOA CÔNG NGHỆ THÔNG TIN
BỘ MÔN CÔNG NGHỆ PHẦN MỀM
HỆ CHÍNH QUI
MÔN: **KĨ THUẬT LẬP TRÌNH**

HDTH – TUẦN 02
CẤU TRÚC & MẢNG CẤU TRÚC

- NGUYỄN LÊ HOÀNG DŨNG

TP.HCM, ngày 08 tháng 03 năm 2018

MỤC LỤC

1 Qui định

Qui tắc bố trí project (1 solution, n projects: Bai01, Bai02, …)

1. **Tên solution** có dạng **MSSV_TuanXX**, trong đó XX: 01, 02, 03,… là số thứ tự tuần.
2. **Tên project** là Bai01, Bai02, Bai03,…

2 Bài tập

2.1 Bài 01 – Mảng các điểm trong mặt phẳng Oxy

2.1.1 Đề bài

Viết chương trình, thực hiện các yêu cầu sau:

1. Nhập vào mảng các điểm (hoành độ x, tung độ y) trong mặt phẳng Oxy
2. Xuất mảng các điểm ra màn hình
3. Nhập vào tọa độ điểm A(x,y). Tìm điểm trong mảng cách xa điểm A nhất, nếu có nhiều điểm cùng xa A nhất thì chỉ trả về 1 điểm trong số đó.
4. Liệt kê các điểm có tọa độ dương trong mảng
5. Sắp xếp mảng các điểm tăng dần theo khoảng cách từ điểm đến gốc tọa độ O

2.1.2 Hướng dẫn

- Mở Visual Studio, tạo project với các thông số sau:
  - Ngôn ngữ: Visual C++
  - Template: Windows 32 Console Application
  - Tên project: Bai01
  - Tên solution: MSSV_Tuan02
  - Loại project: Empty project
- Với project này, ta sẽ tạo 5 file:
  - File Diem.h: chứa khai báo struct Diem và các hàm liên quan đến điểm
  - File Diem.cpp: chứa cài đặt các hàm liên quan đến điểm
  - File MangDiem.h: chứa khai báo các hàm liên quan đến mảng điểm
  - File MangDiem.cpp: chứa cài đặt các hàm liên quan đến mảng điểm
  - File main.cpp: chứa hàm main
- Với câu 1, nhập mảng điểm ta cần làm các việc sau:
  - Viết hàm nhập mảng điểm
  - Trong hàm nhập mảng điểm sẽ gọi hàm nhập điểm
  - Do đó, cần khai báo struct Diem
- File Diem.h: khai báo struct Diem và hàm nhập thông tin 1 điểm:

```
#ifndef _DIEM_H_
#define _DIEM_H_
#include<stdio.h>
struct Diem
{
    float x, y;
```

```
};
void NhapDiem(Diem &d);
#endif
```

- File Diem.cpp: cài đặt hàm nhập thông tin 1 điểm

```
void NhapDiem(Diem &d)
{
    // B1: Nhap hoanh do x
    printf("Nhap x: ");
    scanf("%f",&d.x);
    // B2: Nhap tung do y
    printf("Nhap y: ");
    scanf("%f",&d.y);
}
```

- File MangDiem.h: khai báo hàm nhập 1 mảng các điểm

```
#ifndef _MANGDIEM_H_
#define _MANGDIEM_H_
#include "Diem.h"
void NhapMangDiem(Diem a[], int &n);
#endif
```

- File MangDiem.cpp: cài đặt hàm nhập 1 mảng các điểm

```
void NhapMangDiem(Diem a[], int &n)
{
    // B1: Nhap so phan tu cua mang
    printf("Nhap n: ");
    scanf("%d",&n);
    // B2: Nhap gia tri cho tung diem trong mang
    for(int i=0;i<n;i++)
    {
        printf("Nhap diem thu a[%d]",i);
        // B2.1: Goi ham NhapDiem
        NhapDiem(a[i]);
    }
}
```

- File main.cpp: chứa hàm main, khai báo mảng điểm và gọi hàm nhập mảng. Build #1, sửa lỗi, chạy thử chương trình

```
#include "Diem.h"
void main()
{
    Diem a[100];
    int n;
    printf("Nhap mang cac diem, mang a\n");
    NhapMangDiem(a,n);
}
```

- Với yêu cầu 2: xuất mảng các điểm ra màn hình
  - Viết hàm xuất mảng các điểm. Hàm này sẽ gọi:
    - Hàm xuất thông tin 1 điểm ra màn hình
- File Diem.h: khai báo thêm hàm xuất điểm (ngay trước #endif)

```
void XuatDiem(Diem d);
```

- File Diem.cpp: cài đặt hàm xuất điểm

```
void XuatDiem(Diem d)
{
    // B1: Xuat x va y
    printf("%0.3f,%0.3f",d.x,d.y);
}
```

- File MangDiem.h: khai báo hàm xuất mảng điểm

```
void XuatMangDiem(Diem a[], int n);
```

- File MangDiem.cpp: cài đặt hàm xuất mảng điểm

```
void XuatMangDiem(Diem a[], int n)
{
    // B1: Duyet tu dau den cuoi mang
    for(int i=0;i<n; i++)
    {
        printf("Diem a[%d]",i);
        // B1.1: Goi ham XuatDiem XuatDiem(a[i]);
        printf("\n");
    }
}
```

- File main.cpp: gọi thêm hàm xuất mảng điểm, build #2, sửa lỗi, chạy thử

```
void main()
{
   Diem a[100];
   int n;
   printf("Nhap mang cac diem, mang a\n");
   NhapMangDiem(a,n);
   printf("Mang a vua nhap\n");
   XuatMangDiem(a,n);
 }
```

- Với yêu cầu 3, ta viết hàm tìm phần tử trong mảng xa A nhất:
  - Hàm này sẽ gọi hàm tính khoảng cách từ a[i] đến A để xem a[i] nào xa nhất
- Do đó, file Diem.h: khai báo hàm tính khoảng cách 2 điểm

```
float TinhKhoangCach(Diem a, Diem b);
```

- File Diem.cpp: cài đặt hàm tính khoảng cách

```
float TinhKhoangCach(Diem a, Diem b)
```

```
{
   return sqrt((b.x-a.x)*(b.x-a.x)+(b.y-a.y)*(b.y-a.y));
}
```

- File MangDiem.h: khai báo hàm tìm điểm cách xa nhất

```
Diem TimXaNhat(Diem a[], int n, Diem A);
```

- File MangDiem.cpp: cài đặt hàm tìm điểm cách xa nhất

```
Diem TimXaNhat(Diem a[], int n, Diem A)
{
   // B1: Gia su a[0] la diem xa A nhat
   Diem kq=a[0];
   // B2: Tinh khoang cach tu a[0] den A
   float kc=TinhKhoangCach(a[0],A);
   // B3: Duyet tu dau den cuoi mang
   for(int i=1;i<n;i++)
   {
       // B3.1: Tinh khoang cach tu a[i] den A
       float kc2=TinhKhoangCach(a[i],A);
       // B3.2: Neu kc2>kc
       // cap nhat kq va kc
       if(kc2>kc)
       {
          kq=a[i];
          kc=kc2;
       }
   }
   return kq;
}
```

- File main.cpp: gọi hàm, build #3, sửa lỗi, chạy thử

```
Diem A;
printf("Nhap diem A\n");
NhapDiem(A);
Diem x=TimXaNhat(a,n,A);
printf("Diem trong mang xa A nhat la: ");
XuatDiem(x);
printf("\n");
```

- Gợi ý cho câu 4:

```
void LietKeToaDoDuong(Diem a[], int n)
{
     // B1: Duyet tu dau den cuoi mang
        // B1.1: Neu a[i].x>0 va a[i].y>0
           // B1.1.1: Goi ham XuatDiem a[i]
}
```

- Gợi ý cho câu 5:

```
void SapTangTheoKhoangCach(Diem a[], int n)
```

```
{
    // B1: Khai bao diem O
    // ,gan O.x: 0
    // ,gan O.y: 0
    // B2: i duyet tu 0 den n-2
        // B2.1: j duyet tu i+1 den n-1
        // B2.1.1: Neu khoang cach (a[i], O) > khoang cach (a[j], 0)
            // B2.1.1.1: Hoan vi a[i] va a[j]
}
```

2.2 Bài 02 – Mảng các tam giác

2.2.1 Đề bài

Viết chương trình thực hiện các chức năng sau:

1. Nhập vào mảng các tam giác
2. Xuất mảng các tam giác ra màn hình
3. Tìm tam giác có chu vi lớn nhất
4. Tính tổng diện tích các tam giác
5. Đếm các tam giác có diện tích lớn hơn diện tích trung bình trong mảng
6. Sắp xếp mảng các tam giác giảm dần theo chu vi

2.2.2 Hướng dẫn

- Để làm được bài này, chúng ta cần 7 file:
    - File Diem.h: copy từ bài 1 qua xài (ai rảnh thì code lại cũng được)
    - File Diem.cpp: copy từ bài 1 qua xài (ai rảnh thì code lại cũng được)
    - File TamGiac.h: khai báo struct TamGiac và các hàm liên quan đến tam giác

```
#ifndef _TAMGIAC_H_
#define _TAMGIAC_H_
#include "Diem.h"
struct TamGiac
{
   Diem A, B, C;
};
#endif
```

    - File TamGiac.cpp: cài đặt các hàm liên quan đến tam giác
    - File MangTamGiac.h: khai báo các hàm liên quan đến mảng tam giác

```
#ifndef _MANGTG_H_
#define _MANGTG_H_
#include "TamGiac.h"
#endif
```

    - File MangTamGiac.pp: cài đặt các hàm liên quan đến mảng tam giác
    - File main.cpp: chứa hàm main
- Yêu cầu 1: hàm nhập mảng các tam giác

- Gọi hàm nhập 1 tam giác
  - Gọi hàm nhập điểm 3 lần để nhập giá trị cho đỉnh A, B và C

```
void NhapTamGiac(TamGiac &t)
{
    printf("Nhap dinh A\n");
    NhapDiem(t.A);
    printf("Nhap dinh B\n");
    NhapDiem(t.B);
    printf("Nhap dinh C\n");
    NhapDiem(t.C);
}
void NhapMangTamGiac(TamGiac a[], int &n)
{
    // B1: Nhap so phan tu, n
    // B2: Duyet tu dau den cuoi mang
    for(int i=0;i<n;i++)
    {
        printf("Nhap tam giac a[%d]\n",i);
        // B2.1: Goi ham NhapTamGiac a[i]
        NhapTamGiac(a[i]);
    }
}
```

- Yêu cầu 2: hàm xuất mảng các tam giác
  - Gọi hàm xuất 1 tam giác
    - Gọi hàm xuất điểm 3 lần để xuất giá trị cho đỉnh A, B, C

```
void XuatTamGiac(TamGiac t)
{
    printf("Dinh A: ");
    XuatDiem(t.A);
    printf("\n");
    printf("Dinh B: ");
    XuatDiem(t.B);
    printf("\n");
    printf("Dinh C: ");
    XuatDiem(t.C);
    printf("\n");
}
void XuatMangTamGiac(TamGiac a[], int n)
{
    for(int i=0;i<n;i++)
    {
        printf("Tam giac a[%d]: ");
        // Goi ham XuatTamGiac a[i]
        printf("\n");
    }
}
```

- Yêu cầu 3: hàm tìm tam giác có chu vi lớn nhất
  - Gọi hàm tính chu vi 1 tam giác
    - Gọi hàm tính khoảng cách 3 lần để tính khoảng cách 3 cạnh AB, BC, CA

```
float TinhChuVi(TamGiac t)
{
    float AB=TinhKhoangCach(t.A, t.B);
    float BC=TinhKhoangCach(t.B, t.C);
    float CA=TinhKhoangCach(t.C, t.A);
    return AB+BC+CA;
}
TamGiac TimChuViNhoNhat(TamGiac a[], int n)
{
    // B1: Gia su kq = a[0] la tam giac co chu vi nho nhat
    // B2: Duyet tu dau den cuoi mang
    // B2.1: Neu chu vi (a[i]) > chu vi (kq)
    // Cap nhat kq=[i]
}
```

- Yêu cầu 4: hàm tính tổng diện tích của các tam giác
  - Gọi hàm tính diện tích 1 tam giác
    - Gọi hàm tính khoảng cách 3 lần để tính khoảng cách 3 cạnh AB, BC, CA

```
float TinhDienTich(TamGiac t)
{
    float AB=TinhKhoangCach(t.A, t.B);
    float BC=TinhKhoangCach(t.B, t.C);
    float CA=TinhKhoangCach(t.C, t.A);
    // Tinh nua chu vi
    float p=(AB+BC+CA)/2;
    return sqrt(p*(p-AB)*(p-BC)*(p-CA));
}
float TinhTongDienTich(TamGiac a[], int n)
{
    // B1: tong = 0
    // B2: Duyet tu dau den cuoi mang
    // B2.1: Cong tong voi dien tich a[i]
}
```

- Yêu cầu 5: hàm đếm số tam giác có diện tích lớn hơn diện tích trung bình
  - Gọi hàm tính tổng diện tích (câu 4)
  - Gọi hàm tính diện tích cho từng a[i] của tam giác

```
int DemTamGiacLonHonDTichTBinh(TamGiac a[], int n)
{
    // B1: tb=TinhTongDienTich(a,n)/n;
```

```
    // B2: Gan dem: 0
    // B3: Duyet tu dau den cuoi mang
    // B3.1: Neu dien tich(a[i]) > tb
    // B3.1.1: dem ++
}
```

- Yêu cầu 6: hàm sắp xếp các tam giác giảm dần theo chu vi

```
void SapGiamTheoChuVi(TamGiac a[], int n)
{
    // B1: i duyet tu 0 den n-2
    // B1.1: j duyet tu i+1 den n-1
    // B1.1.1: Neu chu vi (a[i], 0) < chu vi (a[j], 0)
    // B1.1.1.1: Hoan vi a[i] va a[j]
}
```

2.3 Bài 03 – Mảng các đường tròn

2.3.1 Đề bài

Viết chương trình thực hiện các chức năng sau:

1. Nhập vào mảng các đường tròn
2. Xuất mảng các đường tròn ra màn hình
3. Tìm đường tròn có chu vi lớn nhất
4. Tính tổng diện tích các đường tròn
5. Đếm các đường tròn có diện tích lớn hơn diện tích trung bình trong mảng
6. Sắp xếp mảng các đường tròn giảm dần theo chu vi

2.3.2 Hướng dẫn

```
#ifndef _DUONGTRON_H_
#define _DUONGTRON_H_
#include "Diem.h"
struct DuongTron
{
   Diem tam;
   float bankinh;
};
#endif
```

2.4 Bài 04 – Vector

2.4.1 Đề bài

Viết chương trình thực hiện các chức năng sau:

1. Khởi tạo vector rỗng, có n thành phần
2. Khởi tạo vector có n thành phần, mỗi thành phần có giá trị 0
3. Khởi tạo vector từ mảng a, n phần tử cho trước
4. Khởi tạo vector từ 1 vector khác

5. Nhập giá trị cho vector
6. Xuất vector ra màn hình
7. Cộng 2 vector: nếu 2 vector cùng n thành phần thì cộng từng thành phần lại với nhau. Ngược lại, báo lỗi
8. Nhân vector với một số thực
9. Tích vô hướng 2 vector: nếu 2 vector cùng n thành phần thì nhân từng thành phần lại với nhau. Ngược lại, báo lỗi

2.4.2 Hướng dẫn

Bài này cần 3 file: HTThanhVector.h, HTThanhVector.cpp và main.cpp

```
#ifndef _HTTHANHVECTOR_H_
#define _HTTHANHVECTOR_H_
#include<stdio.h>
struct Vector
{
      float a[100];
      int n;
};
#endif
```

1. Khởi tạo vector rỗng, có n thành phần

```
void KhoiTaoRong(Vector &v)
{
   v.n=0;
}
```

2. Khởi tạo vector có n thành phần, mỗi thành phần có giá trị 0

```
void KhoiTaoN(Vector &v, int m)
{
    v.n=m;
    for(int i=0;i<v.n;i++)
    {
        v.a[i]=0;
    }
}
```

3. Khởi tạo vector từ mảng a, n phần tử cho trước

```
void KhoiTaoMang(Vector &v, int a[], int m)
{
    v.n=m;
    for(int i=0;i<v.n;i++)
    {
        v.a[i]=a[i];
    }
}
```

4. Khởi tạo vector từ 1 vector khác

```
void KhoiTaoSaoChep(Vector &v, Vector w)
{
    v.n=w.n;
    for(int i=0;i<v.n;i++)
    {
        v.a[i]=w.a[i];
    }
}
```

5. Nhập giá trị cho vector

```
void NhapVector(Vector &v)
{
   // B1: Nhap v.n
   // B2: Duyet tu dau den cuoi mang,
   // nhap gia tri v.a[i]
}
```

6. Xuất vector ra màn hình

```
void XuatVector(Vector v)
{
   // B1: Duyet tu dau den cuoi mang,
   // xuat v.a[i]
 }
```

7. Cộng 2 vector: nếu 2 vector cùng n thành phần thì cộng từng thành phần lại với nhau. Ngược lại, báo lỗi

```
Vector CongVector(Vector v, Vector w)
{
   // B1: Neu v.n khac w.n,
   // bao loi & return
   // B2: Khai bao Vector kq;
   // B3: Gan kq.n: v.n
   // B4: Duyet tu dau den cuoi mang
   // kq.a[i]=v.a[i]+w.a[i]
 }
```

8. Nhân vector với một số thực

```
Vector NhanSoThuc(Vector v, float x)
{
   // B1: Khai bao Vector kq;
   // B2: Gan kq.n: v.n
   // B3: Duyet tu dau den cuoi mang
      // kq.a[i]=v.a[i]*x
 }
```

9. Tích vô hướng 2 vector: nếu 2 vector cùng n thành phần thì nhân từng thành phần lại với nhau. Ngược lại, báo lỗi

```
float TinhVoHuong(Vector v, Vector w)
```

```
{
    // B1: Neu v.n khac w.n,
    // bao loi & return
    // B2: float tong: 0
    // B3: Duyet tu dau den cuoi mang
    // tong+=v.a[i]*w.a[i]
}
```

2.5 Bài tập về con trỏ

Làm lại các Bài tập 1, Bài tập 2, Bài tập 3 bằng cách sử dụng con trỏ cấp phát động.