Trường CĐKT CAO THẮNG
Bộ môn TIN HỌC

Tài liệu Thực hành

CẤU TRÚC DỮ LIỆU và THUẬT TOÁN

Biên soạn:
Nguyễn Thị Thanh Thuận
Vũ Đình Bảo

Họ tên sinh viên: ……

Lớp: ……

2 / 2016

BÀI THỰC HÀNH SỐ 1: TỔNG QUAN

Ngày học: ……. Điểm: ……

Lưu ý: *Sinh viên tự ghi kết quả thực hành bài tập này, tất cả bài tập sau cũng ghi kết quả sau mỗi buổi học:*

- *Đã cài đặt hoàn chỉnh và chạy thử nghiệm đúng ……. lần*
- *Đã cài đặt nhưng khi chạy thử nghiệm có lúc đúng lúc sai*
- *Chương trình bị lỗi/ Chưa cài đặt xong/ Không biết làm …*

Bài 1.1 (1 tiết)

Sinh viên chạy thử chương trình bên dưới, đọc hiểu chương trình, trả lời các câu hỏi:

Các đối số hàm **Nhap** và hàm **Xuat** có gì khác nhau? Tại sao?

…….

Trong hàm **Nhap** phân biệt cách dùng **gets** và **cin**

…….

Trong hàm **Main** vòng lặp **do … while** lặp mấy lần thì dừng

…….

```
#include<iostream>
#include<conio.h>
using namespace std;

struct NhanVien
{
	char HoTen[25];
	int NamSinh;
};

void Nhap(int &n, NhanVien NV[])
{
	cout << "\nNhap so luong: ";
	cin >> n;
	for (int i = 0; i < n; i++)
	{
		cout << "\nNhap ho ten: ";
		fflush(stdin);
		gets(NV[i].HoTen);
		cout << "\nNhap nam sinh: ";
		cin >> NV[i].NamSinh;
	}
}

void Xuat(int n, NhanVien NV[])
```

```
{
	cout << "\nDanh sach nhan vien la: ";
	for (int i = 0; i < n; i++)
	{
		cout << "\nHo ten nguoi thu " << i + 1 << " la: ";
		cout << NV[i].HoTen;
		cout << " - nam sinh: ";
		cout << NV[i].NamSinh;
	}
}
int Tim(int n, NhanVien NV[], char s[])
{
	int i;
	for (i = 0; i < n; i++)
		if (strcmp(NV[i].HoTen,s) == 0) break;
	return i;
}

void main()
{
	int n, kq;
	char ten[25];
	NhanVien NV[50];
	Nhap(n, NV);
	Xuat(n, NV);
	do
	{
		cout << "\nBan muon tim nhan vien co ten gi?: ";
		fflush(stdin);
		gets(ten);
		kq = Tim(n, NV, ten);
		if (kq == n) cout << "Khong co";
		else cout << "NV can tim nam sinh la:" << NV[kq].NamSinh;
		cout << "\nNhan esc de thoat, nhan phim khac de tim tiep";
	} while (getch() != 27);
}
```

Viết thêm hàm **Tim1** có ý nghĩa giống hàm **Tim** thay vòng lặp **for** bằng **while**

Bài 1.2 (1 tiết)

Viết chương trình nhập hồ sơ thí sinh gồm n mẫu tin, mỗi mẫu tin gồm các thông tin: họ tên, tuổi, điểm lý thuyết, điểm thực hành (điểm có 1 số thập phân). Xuất các thông tin đã nhập.

1. Xuất thông tin thí sinh có tuổi lớn hơn 18.
2. Nhập họ tên xuất thông tin thí sinh đầu tiên có họ tên tương ứng.
3. Nhập tuổi xuất thông tin những thí sinh có tuổi tương ứng.
4. Sắp xếp thông tin theo thứ tự điểm trung bình giảm dần.
5. Chương trình lặp liên tục các chọn lựa 1,2,3,4 cho đến khi nhấn phím khác lựa chọn thì dừng.

Bài 1.3 (1 tiết)

Cho dãy a gồm n số nguyên. Tìm tổng lớn nhất trong tất cả các tổng của các dãy con của a

Ví dụ: –5 6 –4 2 3 -7 kết quả tổng dãy con lớn nhất là 7.

Làm bài theo nhiều cách khác nhau với độ phức tạp thuật toán khác nhau

Gợi ý: chúng ta sẽ tính lần lượt từng tổng của các dãy con để tìm ra tổng lớn nhất

-5

-5+6

-5+6+-4

-5+6+-4+2

-5+6+-4+2+3

-5+6+-4+2+3+-7

6

6+-4

6+-4+2

6+-4+2+3

6+-4+2+3+-7

-4

-4+2

-4+2+3

….

Sinh viên ghi lại các câu thông báo lỗi bằng tiếng anh đã gặp trong bài này và dịch sang tiếng viết

BÀI THỰC HÀNH SỐ 2: TÌM KIẾM

Ngày học: ..... Điểm: .....

Bài 2.1 (0,5 tiết) Thử nghiệm chương trình:

```
int LinearSearch1(int a[], int N, int x)
{
	int i = 0;
	while ((i < N) && (a[i] != x)) i++;
	if (i == N)
		return -1; // tìm hết nhưng không có x
	return i;		// a[i] là phần tử có khóa x
}
int LinearSearch2(int a[], int N, int x)
{
	int i = 0; // mảng gồm N phần tử từ a[0]...a[N-1]
	a[N] = x;  // thêm phần tử thứ N
	while (a[i] != x)  i++;
	if (i == N)
		return -1; // tìm hết nhưng không có x
	return i; // tìm thấy x tại vị trí i
}
void main()
{
	int a[8] = {3, 6, 1, 9, 2, 7, 5};
	int N = 7, x;
	cout << "Mang la 3, 6, 1, 9, 2, 7, 5";
	do
	{
		cout << "\n\nNhap so can tim: ";
		cin >> x;

		int c1 = LinearSearch1(a, N, x);
		if (c1 == -1)
			cout << "Da ss " << N * 2 + 1 << "lan. kq=khong co";
		else
			cout << "\nDa ss " << (c1 + 1) * 2 << "lan. kq=co o vi
tri " << c1;

		cout << "\nDUNG LINH CANH";
		int c2 = LinearSearch2(a, N, x);
		if (c2 == -1)
			cout << "\nDa ss " << N + 1 << "lan. kq=khong co";
		else
			cout << "\nDa ss " << c2 + 1 << "lan. kq=co o vi tri "
<< c2;

		cout << "\n Nhan Esc de dung, phim khac de tim tiep ";
	} while (getch() != 27);
}
```

Hãy lần lượt tìm 6, 5, 8
Thay giá trị mảng khác, chú ý số lượng phần tử.
Nếu có 2 số cùng giá trị thì tìm thấy ở vị trí nào?.....

Bài 2.2 (1 tiết)
Viết chương trình:

a. Khởi tạo mảng có giá trị sau: **1 2 3 5 6 7 9**
b. Thử nghiệm 2 hàm tìm kiếm nhị phân đã học, ví dụ

```
mang 1,2,3,5,6,7,9

nhap so can tim 5
tim thay o vi tri 3
Nhan Esc de dung, phim khac de tim tiep

nhap so can tim 12
k co
Nhan Esc de dung, phim khac de tim tiep

nhap so can tim 1
tim thay o vi tri 0
Nhan Esc de dung, phim khac de tim tiep
```

c. Viết thêm vào hàm **BinarySearch** để xuất thêm giá trị **l,r,m** lần lượt là bao nhiêu khi tìm. Kiểm chứng lại các dãy số đã làm trong giờ lý thuyết.

```
mang 1,2,3,5,6,7,9

nhap so can tim 5
l=0 r=6 m=3tim thay o vi tri 3
Nhan Esc de dung, phim khac de tim tiep

nhap so can tim 12
l=0 r=6 m=3
l=4 r=6 m=5
l=6 r=6 m=6
l=7 r=6k co
Nhan Esc de dung, phim khac de tim tiep

nhap so can tim 1
l=0 r=6 m=3
l=0 r=2 m=1
l=0 r=0 m=0tim thay o vi tri 0
Nhan Esc de dung, phim khac de tim tiep

nhap so can tim 6
l=0 r=6 m=3
l=4 r=6 m=5
l=4 r=4 m=4tim thay o vi tri 4
Nhan Esc de dung, phim khac de tim tiep
```

Bài 2.3 (1,5 tiết)
Viết chương trình:

a. Nhập mảng 1 chiều các số nguyên, xuất mảng
b. Viết hàm đếm xem trong dãy có bao nhiêu số bằng x. (nếu không tìm thấy thì in ra câu: ”Khong tim thay”)
c. Viết hàm xuất các vị trí của x trong mảng (nếu không tìm thấy thì in ra câu: ”Khong tim thay”).
d. Tạo menu chọn lựa câu a,b,c. chương trình lặp liên tục cho đến khi nhấn ESC thì dừng

```
void input(int a[], int N)
{
	for (int i = 0; i < N; i++)
		cin >> a[i];
}
void output(int a[], int N)
{
	for (int i = 0; i < N; i++)
		cout << a[i] << " ";
}
void main()
{
	int a[10], N;
	cout << "Nhap so luong phan tu: ";
	cin >> N;
	input(a, N);
	cout << "\nMang vua nhap la: ";
	output(a, N);
}
```

Ghi lại hàm câu b hoặc c sau khi đã chạy thử chương trình đúng

BÀI THỰC HÀNH SỐ 3: SẮP XẾP

Straight Interchange Sort, Straight Selection Sort

Ngày học: …….. Điểm: ……

Bài 3.1 (1 tiết)
Viết chương trình:

- Nhập mảng 1 chiều các số nguyên, xuất mảng
- Sắp mảng tăng dần bằng phương pháp **Straight Interchange Sort**
- Sắp mảng tăng dần bằng phương pháp **Straight Selection Sort**

Trình bày kết quả rõ ràng, dễ nhìn, đẹp.

Bài 3.2 (0,5 tiết)
Hiệu chỉnh lại **Bài 3.1** để xuất mảng sau mỗi lần hoán vị. Ví dụ:

```
Mang vua nhap la: 3 6 2 9 1

InterchangeSort:
2 6 3 9 1
1 6 3 9 2
1 3 6 9 2
1 2 6 9 3
1 2 3 9 6
1 2 3 6 9

SelectionSort:
1 6 2 9 3
1 2 6 9 3
1 2 3 9 6
1 2 3 6 9
```

Ghi lại hàm sắp xếp có thêm lệnh xuất mảng sau mỗi lần hoán vị

Bài 3.3 (1,5 tiết)
Viết chương trình tạo một menu với các mục chọn như sau:

- Nhập mảng 1 chiều các số nguyên, xuất mảng
- Nhập số nguyên N, tạo mảng 1 chiều gồm N số nguyên ngẫu nhiên từ 0-9, xuất mảng
- Sắp mảng tăng dần bằng phương pháp **Straight Interchange Sort**
- Sắp mảng tăng dần bằng phương pháp **Straight Selection Sort**

Chương trình lặp liên tục cho đến khi nhấn ESC thì dừng.
Trình bày kết quả rõ ràng, dễ nhìn, đẹp.

Gợi ý:

```
#include<time.h>
void create(int a[], int N)
{
	srand(time(0));
	for (int i = 0; i < N; i++)
		a[i] = rand() % 100 + 1;
}
```

BÀI THỰC HÀNH SỐ 4: SẮP XẾP (tt)

Kiểm tra thực hành - Quick Sort

Ngày học: ……. Điểm: …….

Bài 4.1 (1 tiết)
Tiếp tục **Bài 3.2**

- Sau khi sắp xếp cho biết **Số lần hoán vị, Số lần so sánh** đã thực hiện để sắp xếp
- Viết thêm hàm sắp giảm dần theo các phương pháp trên

Ví dụ
Nhập mảng có giá trị sau: **3 6 1 9 2 7 5**

- Số nghịch thế:
- Dùng **Straight Interchange Sort**
  - Số lần hoán vị:
  - Số lần so sánh:
- Dùng **Straight Selection Sort**
  - Số lần hoán vị:
  - Số lần so sánh:
- Mảng sau khi sắp:

Tạo mảng gồm các số ngẫu nhiên với N lớn (N=100, 1000), so sánh độ phức các thuật toán sắp xếp.

Bài 4.2 (1 tiết)
Tiếp tục **Bài 3.2** thêm phương pháp **Quick Sort**

Bài 4.3
Viết lại bài giải hoàn chỉnh của bài kiểm tra

BÀI THỰC HÀNH SỐ 5

Ôn tập Tìm kiếm và Sắp xếp

Ngày học: …….. Điểm: …….

Bài 5.1 (1 tiết)

Viết chương trình nhập vào mảng 2 chiều m dòng n cột

a. Sắp tăng dần các phần tử trên mỗi dòng
b. Sắp tăng dần các phần tử trên mỗi cột
c. Sắp tăng dần các phần tử theo chiều từ trái qua phải và từ trên xuống dưới.
d. Tạo menu chọn lựa câu a,b,c. chương trình lặp liên tục cho đến khi nhấn ESC thì dừng

Bài 5.2 (2 tiết)

Nhập vào danh sách học sinh gồm các thông tin: MSSV, họ tên, điểm trung bình.

a. Tìm **những sinh viên** có điểm trung bình cao nhất, thấp nhất.
b. Sắp tăng dần theo điểm trung bình bằng nhiều cách sắp xếp.
c. Sắp tăng dần theo MSSV
d. Tạo menu chọn lựa câu a,b,c. chương trình lặp liên tục cho đến khi nhấn ESC thì dừng

Ghi lại hàm câu a,b

BÀI THỰC HÀNH SỐ 6: DSLK

Thêm và Duyệt

Ngày học: ..... Điểm: .....

```
typedef struct node
{
       int info;
       node *next;
 };

typedef struct list
{
      node *head;
      node *tail;
};

node* GetNode(int x)
{
      node *p;
      p = new node;
      if (p == NULL)
      {
            cout << "Khong du bo nho";
            exit(1); // thoát khỏi hàm GetNode
      }
      p->info = x;
      p->next = NULL;
      return p;
}

void InsertHead(list &l, int x)
{
      node *p;
      p = GetNode(x) ;
      if (p == NULL)
      {
            cout << " Khong tao duoc nut !";
            exit(1);
      }
      if (l.head == NULL)  //  danh sách đơn rỗng
            l.head = l.tail = p;
      else
      {
            p->next = l.head;
            l.head = p;
      }
}
```

```
void OutPut(list &l)
{
	node *p;
	p = l.head;
	while (p != NULL)
	{
		cout << p->info << " ";
		p = p->next;
	}
}
void main()
{
	list l;
	l.head = l.tail = NULL;
	int x;
	cout << "Nhap noi dung nut, nhap 0 de dung: "; cin >> x;
	while (x != 0)
	{
		InsertHead(l, x);
		cout << "Nhap noi dung nut, nhap 0 de dung: "; cin >> x;
	}
	OutPut(l);
}
```

Khi chạy chương trình trên, ví dụ nếu nhập vào 3 4 2 7 6 0 thì sẽ xuất ra gì?

Ý nghĩa chương trình trên?

Có cần thay đổi gì không?

Viết tiếp các yêu cầu sau

a. Tính tổng các phần tử trong danh sách
   Ví dụ trên sẽ xuất kết quả là 12
b. Đếm số nút của danh sách
   Ví dụ trên sẽ xuất kết quả là 5
c. Kiểm tra xem x có trong danh sách hay không
   Ví dụ trên nếu nhập x=2 sẽ xuất kết quả là ***có***
   x=5 sẽ xuất kết quả là ***không có***
d. Tìm phần tử lớn nhất trong danh sách
   Ví dụ trên sẽ xuất kết quả là 7
e. Đếm xem trong danh sách có bao nhiêu số nguyên tố
   Ví dụ trên sẽ xuất kết quả là ***3,7 là 2 số nguyên tố trong dslk***

BÀI THỰC HÀNH SỐ 7: DSLK (tt)
Xóa và Duyệt

Ngày học: ..... Điểm: .....

Viết chương trình nhập, xuất danh sách liên kết các số nguyên.

a. Đảo ngược danh sách

b. Xoá một phần tử tại vị trí thứ k trong danh sách

c. Xoá phần tử có giá trị x trong danh sách

d. Sắp xếp danh sách tăng dần theo nhiều cách.

e. Chèn một phần tử có giá trị x vào vị trí thứ k của danh sách

f. Tách danh sách thành 2 danh sách: một danh sách chứa toàn số chẵn, một danh sách chứa toàn số lẻ.

BÀI THỰC HÀNH SỐ 8: DSLK (tt)

DSLK Sinh viên - Kiểm tra thực hành

Ngày học: ..... Điểm: .....

Bài 8.1

Ứng dụng danh sách liên kết, xây dựng chương trình quản lý sinh viên, thông tin sinh viên cần: MSSV, Điểm (số nguyên từ 0-10), Ghi chú

a. Nhập xuất danh sách sinh viên.
b. In ra danh sách những sinh viên có điểm >=8.
c. Tìm sinh viên theo MSSV.
d. Hủy sinh viên có điểm <2.
e. Sắp xếp danh sách theo thứ tự điểm giảm dần.
f. Thêm 1 sinh viên mới vào danh sách đã có thứ tự điểm giảm dần sao cho thứ tự vẫn đúng.

Bài 8.2

Cho hai danh sách liên kết gồm các nút có khoá là các số nguyên đã được sắp tăng dần. Viết chương trình nối hai danh sách trên để tạo ra một danh sách liên kết cũng được sắp theo thứ tự tăng dần.

Bài 8.3

Viết lại bài giải hoàn chỉnh của bài kiểm tra

BÀI THỰC HÀNH SỐ 9: DSLK (tt)

Stack

Ngày học: …….. Điểm: ……

Bài 9.1
Viết chương trình nhập số nguyên trong hệ thập phân, xuất lại số đó dạng hệ nhị phân và thập lục phân

Hệ 10	Hệ 2	Hệ 16
9	1001	9
10	1010	A
14	1110	E

Bài 9.2 (Sưu tập từ Thầy Phù Khắc Anh)
Nhập vào một biểu thức toán học, bao gồm các số nguyên và các phép toán +,-,*,/, (,)

Hãy tính giá trị biểu thức trên theo đúng độ ưu tiên toán học (ưu tiên trong ngoặc, ưu tiên nhân chia trước, cộng trừ sau).

Hướng dẫn:
B1: Nhập biểu thức.
B2: Chuyển đổi biểu thức từ dạng trung tố sang hậu tố
B3: Tính giá trị biểu thức từ stack đã xây dựng ở B2

BÀI THỰC HÀNH SỐ 10: DSLK (tt)

Queue

Ngày học: ..... Điểm: .....

Viết chương trình ghép các cặp khiêu vũ trong một buổi tiệc, qui tắc ghép cặp phải gồm 1 nam và 1 nữ theo thứ tự vào trước ghép trước. Ví dụ:

```
NHAP THONG TIN NGUOI THAM GIA, NAM NHAP 0, NU NHAP 1, Ten=0 se dung nhap lieu
Nguoi thu:1
Ten:La An
Gt:0

Nguoi thu:2
Ten:Nguyen
Gt:0

Nguoi thu:3
Ten:Ngoc Lan
Gt:1

Nguoi thu:4
Ten:Le Van
Gt:0

Nguoi thu:5
Ten:Yen
Gt:1

Nguoi thu:6
Ten:Nam
Gt:0

Nguoi thu:7
Ten:0

Cap 1 la: La An va Ngoc Lan
Cap 2 la: Nguyen va Yen
So nam con lai la:Le Van Nam
```

BÀI THỰC HÀNH SỐ 11: DSLK (tt)

Ôn tập - Kiểm tra thực hành

Ngày học: …….. Điểm: …….

Viết lại bài giải hoàn chỉnh của bài kiểm tra

Viết chương trình:

Tạo dslk gồm các số 5, 2, 7, 3, 8.
Thêm số x vào sau số y.
Xuất tổng các số chẳn trong dslk.

BÀI THỰC HÀNH SỐ 12: Cây

Cây nhị phân tìm kiếm

Ngày học: ..... Điểm: .....

Chương trình trình sau có ý nghĩa gì?

```
typedef struct TNode
{
	int Key;
	TNode *pLeft, *pRight;
};
typedef TNode* Tree;
int InsertNode(Tree &T, int x)
{
	if (T != NULL)
	{
		if (T->Key == x) return 0;
		if (T->Key > x) return InsertNode(T->pLeft, x);
		else  return InsertNode(T->pRight, x);
	}
	T = new TNode;
	if (T == NULL) return -1;
	T->Key = x;
	T->pLeft = T->pRight = NULL;
	return 1;
}
void NLR(Tree T)
{
	if (T != NULL)
	{
		cout << T->Key << " ";
		NLR(T->pLeft);
		NLR(T->pRight);
	}
}

void main()
{
	Tree T = NULL;
	int x;
	cout << "Nhap gia tri nut, nhap 0 de dung"; cin >> x;
	while (x != 0)
	{
		InsertNode(T, x);
		cout << "Nhap gia tri nut, nhap 0 de dung"; cin >> x;
	}
	cout << "\nDuyet cay theo thu tu NLR: ";
	NLR(T);
}
```

Viết tiếp chương trình thực hiện các thao tác trên cây NPTK

a. Tìm kiếm nút có giá trị k.
b. Huỷ nút có giá trị k, duyệt cây xem kết quả
c. Cải tiến chương trình (chương trình thực hiện lặp nhiều lần, giá trị nút ngẫu nhiên)

Viết các lời gọi hàm trong `void main()`

BÀI ĐỌC THÊM

Cây nhị phân tìm kiếm cân bằng

Ngày học: ..... Điểm: .....

Bài 1: Hãy định nghĩa cấu trúc dữ liệu cho cây AVL
Hướng dẫn:

```
struct AVLNode
{
	char balanceFactor;   // Chỉ số cân bằng
	<Kiểu dữ liệu> key;
	AVLNode *pLeft, *pRight;
}
typedef AVLNode *AVLTree;
```

Bài 2: Viết hàm quay đơn Left – Left

```
void RotateLL(AVLTree &T)
{
	AVLNode* T1 = T->pLeft;
	T->pLeft = T1->pRight;
	T1->pRight = T;
	switch (T1->balFactor)
	{
		case LH:
			T->balFactor = EH;
			T1->balFactor = EH;
			break;
		case EH:
			T->balFactor = LH;
			T1->balFactor = RH;
			break;
	}
	T = T1;
}
```

Bài 3: Viết hàm quay kép Left – Right

```
// Hàm quay kép Left-Right
void RotateLR(AVLTree &T)
{
	AVLNode* T1 = T->pLeft;
	AVLNode* T2 = T1->pRight;
	T->pLeft = T2->pRight;
	T2->pRight = T;
	T1->pRight = T2->pLeft;
	T2->pLeft = T1;
	switch (T2->balFactor)
	{
		case LH:
			T->balFactor = RH;
			T1->balFactor = EH;
```

```
                break;
            case EH:
                T->balFactor = EH;
                T1->balFactor = EH;
                break;
            case RH:
                T->balFactor = EH;
                T1->balFactor = LH;
                break;
        }
        T2->balFactor = EH;
        T = T2;
}
```

Bài 4: Viết hàm cân bằng khi cây lệch về bên trái:

```
// Hàm cân bằng khi cây lệch về bên trái
int BalanceLeft(AVLTree &T)
{
        AVLNode* T1 = T->pLeft;
        switch (T1->balFactor)
        {
            case LH:
                RotateLL(T);
                return 2;
            case EH:
                RotateLL(T);
                return 1;
            case RH:
                RotateLR(T);
                return 2;
        }
        return 0;
}
```

Bài 5: Viết hàm tạo ra cây có dạng như hình bên dưới. Hãy cân bằng lại cây này.