BÀI TẬP 2

CHUYÊN ĐỀ TỔ CHỨC DỮ LIỆU
KÌ 2 2022-2023, HỆ ĐÀO TẠO TỪ XA

MSSV: 21880159

Họ và Tên: Nguyễn Hữu Vinh

1. (2 đ). Bài tập 4.1.6

Mã nguồn C++:

```
//Hàm sort
void sort(double arr[], int n) {
    for (int i = 0; i < n - 1; i++) {
        for (int j = i + 1; j < n; j++) {
            if (arr[i] > arr[j]) {
                double temp = arr[i];
                arr[i] = arr[j];
                arr[j] = temp;
            }
        }
    }
}
//Hàm tính trung vị
double median(double arr[], int n) {
    sort(arr, n);

    if (n % 2 == 0) {
        return (arr[n / 2 - 1] + arr[n / 2]) / 2.0;
    }
    else {
        return arr[n / 2];
    }
}
```

2. (2 đ). Bài tập 4.2.6.

Mã nguồn C++:

```
//Ma trận
int matrix[1001][1001] = { 0 };

//Khởi tạo
void initialize()
{
	matrix[0][0] = 1;
	for (int i = 1; i < 1001; i++) {
		matrix[i][0] = 1; //nCr = 1 khi r = 0
		for (int j = 1; j < i + 1; j++) {
			matrix[i][j] = (matrix[i - 1][j - 1] + matrix[i - 1][j]);
		}
```

```
        }
}

//Tính tổ hợp nCr
int nCr(int n, int r)
{
        return maxtrix[n][r];
}
```

3. (2 đ). Bài tập 4.4.1.

1.5.6.

Mã nguồn C++:

```
bool isDigit(char c) {
    return (c >= '0' && c <= '9');
}

int count(char arr[]) {
    int count = 0;
    int i = 0;

    for (int i = 0; i < strlen(arr); i++) {
        if (!isDigit(arr[i])) {
            throw exception("Khong phai la so!");
        }
    }
    if (strlen(arr) > 100) {
        throw exception("Vuot qua 100 chu so!");
    }

    while (arr[i] == '0') {
        i++;
    }

    while (arr[i] != '\0' && i < 100) {
        if (isDigit(arr[i])) {
            count++;
        }
        i++;
    }

    return count;
}
```

1.5.7.

Mã nguồn C++:

```
bool isDigit(char c) {
    return (c >= '0' && c <= '9');
}

int sumDigits(char arr[]) {
    int sum = 0;
    int i = 0;

    for (int i = 0; i < strlen(arr); i++) {
        if (!isDigit(arr[i])) {
            throw exception("Khong phai la so!");
        }
```

```
    }

    if (strlen(arr) > 100) {
        throw exception("Vuot qua 100 chu so!");
    }

    while (arr[i] != '\0') {
        if (isDigit(arr[i])) {
            sum += (arr[i] - '0');
        }
        i++;
    }

    return sum;
}
```

1.5.8.

Mã nguồn C++:

```
bool isDigit(char c) {
    return (c >= '0' && c <= '9');
}

string addCommas(char arr[]) {
    int n = strlen(arr);
    string result = "";

    for (int i = 0; i < strlen(arr); i++) {
        if (!isDigit(arr[i])) {
            throw exception("Khong phai la so!");
        }
    }

    if (strlen(arr) > 100) {
        throw exception("Vuot qua 100 chu so!");
    }

    int k = 0;
    while (arr[k] == '0') {
        k++;
    }

    if (k == strlen(arr)) return "0";

    int count = 0;
    for (int i = n - 1; i >= k; i--) {
        result = arr[i] + result;
        count++;

        if (count % 3 == 0 && i > 0) {
            result = "," + result;
        }
    }

    return result;
}
```

4. (2 đ). Bài tập 4.5.5.

Mã nguồn C++:

```cpp
#include <iostream>
#include <string>

#define M_PI 3.14159265358979323846

using namespace std;

struct Complex {
    double real;
    double imaginary;
};

//Liên hợp
Complex conjugate(Complex c1) {
    Complex result;
    result.real = c1.real;
    result.imaginary = -c1.imaginary;
    return result;
}

//Nghịch đảo số phức
Complex reciprocal(Complex c) {
    Complex result;
    double denominator = c.real * c.real + c.imaginary * c.imaginary;
    result.real = c.real / denominator;
    result.imaginary = -c.imaginary / denominator;
    return result;
}

//Cộng
Complex add(Complex c1, Complex c2) {
    Complex result;
    result.real = c1.real + c2.real;
    result.imaginary = c1.imaginary + c2.imaginary;
    return result;
}

//Trừ
Complex subtract(Complex c1, Complex c2) {
    Complex result;
    result.real = c1.real - c2.real;
    result.imaginary = c1.imaginary - c2.imaginary;
    return result;
}

//Nhân
Complex multiply(Complex c1, Complex c2) {
    Complex result;
    result.real = c1.real * c2.real - c1.imaginary * c2.imaginary;
    result.imaginary = c1.real * c2.imaginary + c1.imaginary * c2.real;
    return result;
}

//Chia
Complex divide(Complex c1, Complex c2) {
    Complex result;
    double denominator = c2.real * c2.real + c2.imaginary * c2.imaginary;
    result.real = (c1.real * c2.real + c1.imaginary * c2.imaginary) / denominator;
```

```
    result.imaginary = (c1.imaginary * c2.real - c1.real * c2.imaginary) /
denominator;
    return result;
}


// Tính module của số phức
double modulus(Complex c) {
    return sqrt(c.real * c.real + c.imaginary * c.imaginary);
}

// Tính argument của số phức
double argument(Complex c) {
    if (c.real == 0 && c.imaginary == 0) {
        // Số phức không có argument
        return 0;
    }
    else if (c.real == 0) {
        if (c.imaginary > 0) {
            // Số phức nằm trên trục ảo dương
            return  M_PI/ 2;
        }
        else {
            // Số phức nằm trên trục ảo âm
            return -M_PI / 2;
        }
    }
    else {
        double arg = atan(c.imaginary / c.real);
        if (c.real < 0 && c.imaginary >= 0) {
            // Số phức nằm trong góc phần tư thứ hai
            arg += M_PI;
        }
        else if (c.real < 0 && c.imaginary < 0) {
            // Số phức nằm trong góc phần tư thứ ba
            arg -= M_PI;
        }
        return arg;
    }
}

// Lấy phần thực
double real(Complex c) {
    return c.real;
}

// Lấy phần ảo
double imaginary(Complex c) {
    return c.imaginary;
}

// Lấy căn bậc 2
Complex squareRoot(Complex c) {
    double mod = sqrt(c.real * c.real + c.imaginary * c.imaginary);
    double arg = argument(c) / 2;
    Complex result;
    result.real = sqrt(mod) * cos(arg);
    result.imaginary = sqrt(mod) * sin(arg);
    return result;
}
```

```
int main() {

    //Thử giải phương trình 3x^2 + 2x + 5
    cout << "Chuong trinh giai phuong trinh bac 2 ax^2 + bx + c = 0 (co nghiem phuc):"
<< endl;
    cout << "Nhap a: ";
    int a1 = 0;
    cin >> a1;
    cout << "Nhap b: ";
    int b1 = 0;
    cin >> b1;
    cout << "Nhap c: ";
    int c1 = 0;
    cin >> c1;

    //Bieu dien a,b,c dang so phuc
    Complex a = { a1, 0 };
    Complex b = { b1, 0 };
    Complex c = { c1, 0 };
    //Cac tham so
    Complex d = { 4, 0 };  //so 4
    Complex e = { -1, 0 }; //so -1
    Complex f = { 2, 0 };  //so 2
    Complex delta = squareRoot(subtract(multiply(b, b), multiply(d, multiply(a, c))));

    Complex x1 = divide(add(multiply(e,b),delta),multiply(f,a));
    Complex x2 = divide(subtract(multiply(e, b), delta), multiply(f, a));

    string str1 = "";
    if (x1.imaginary != 0) {
        str1 += " + " +  to_string(x1.imaginary) + "i";
    }

    string str2 = "";
    if (x2.imaginary != 0) {
        str2 += " + " + to_string(x2.imaginary) + "i";
    }
    cout << "Phuong trinh " << a.real << "x^2 + " << b.real << "x + " << c.real << " =
0 co hai nghiem la:" << endl;
    cout << "x1 = " << x1.real << str1 << endl;
    cout << "x2 = " << x2.real << str2 << endl;

    return 0;
}
```

Chạy chương trình giải phương trình bậc 2 cho các bài toán:

2x^2 + -7x + 3 = 0

```
Chuong trinh giai phuong trinh bac 2 ax^2 + bx + c = 0 (co nghiem phuc):
Nhap a: 2
Nhap b: -7
Nhap c: 3
Phuong trinh 2x^2 + -7x + 3 = 0 co hai nghiem la:
x1 = 3
x2 = 0.5
```

3x^2 + 2x + 5 = 0

```
Chuong trinh giai phuong trinh bac 2 ax^2 + bx + c = 0 (co nghiem phuc):
Nhap a: 3
Nhap b: 2
Nhap c: 5
Phuong trinh 3x^2 + 2x + 5 = 0 co hai nghiem la:
x1 = -0.333333 + 1.247219i
x2 = -0.333333 + -1.247219i
```

x^2 + -4x + 4 = 0

```
Chuong trinh giai phuong trinh bac 2 ax^2 + bx + c = 0 (co nghiem phuc):
Nhap a: 1
Nhap b: -4
Nhap c: 4
Phuong trinh 1x^2 + -4x + 4 = 0 co hai nghiem la:
x1 = 2
x2 = 2
```