KIỂM TRA GIỮA KỲ - 20CTT3

Thời gian: 60'

Mã đề: 2

Quy định

Sinh viên xem file Practical Midterm Regulations trên Moodle.

Nội dung

1 Câu 1 (4 điểm)

Cho định nghĩa struct `NODE` và `List` như sau:

```
struct NODE {
    string ID;
    float weight;
    int product;
    NODE* pNext;
};
```

```
struct LIST {
    NODE* pHead;
    NODE* pTail;
};
```

Cho ví dụ một phần tập tin `data.txt` lưu trữ thông tin về gà (chic) và vịt (duck) như sau:

```
ID Weight Product
chic001 2.1 30
duck001 2.2 20
duck002 1.5 14
chic002 1.3 24
duck003 2.5 21
```

Trong đó:

- Tập tin không biết trước số dòng
- Mỗi dòng chứa 3 thông tin ngăn cách nhau bởi 1 khoảng trắng gồm:
  - `ID`: Là mã số gia cầm, gồm 7 ký tự, 4 ký đầu đại diện cho loại gia cầm, 3 ký tự sau đại diện cho mã số gia cầm
  - `Weight`: Cân nặng của gia cầm (số thực), đơn vị kilogram
  - `Product`: Sản lượng trứng (số nguyên) theo tháng

Thực hiện các yêu cầu sau:

1. (2 điểm) Đọc nội dung từ tập tin `data.txt` và lưu trữ vào danh sách liên kết theo quy tắc sau:
   - Nếu là gà → thêm vào đầu, nếu là vịt → thêm vào cuối danh sách liên kết
   - Nếu không phải 2 loại trên thì không đưa vào danh sách liên kết

   Sau đó, in danh sách đọc được ra màn hình.

2. (2 điểm) Xóa tất cả gia cầm KHÔNG đạt tiêu chuẩn, biết rằng gà đạt tiêu chuẩn có sản lượng trứng nằm trong khoảng $[25; 32]$ và vịt đạt tiêu chuẩn có sản lượng trứng nằm trong khoảng $[15; 22]$. Sau đó in ra màn hình danh sách cuối cùng theo từng loại. Ví dụ nội dung in ra màn hình như sau:

```
Ga dat chuan:
chic001 2.1 30
Vit dat chuan:
duck001 2.2 20
duck003 2.5 21
```

   Lưu ý: Sinh viên có thể thiết kế nhiều hàm, sau đó gọi chúng trong hàm `Bai01`.

2 Câu 2 (3 điểm)

Cho đoạn mã nguồn thực hiện nhiệm vụ tìm tập con của tập $W = \{w_1, w_2, ..., w_n\}$ gồm n số nguyên dương có tổng bằng với một số nguyên dương t cho trước. Biết rằng $w_1 < w_2 < ... < w_n$.

```
// ...
void SoS(int k, int sum, int total, int w[], int n, bool s[], int t) {
    if (sum == t) {
        for (int i = 0; i < n; i++)
            if (s[i] == true)
                cout << w[i] << " ";

        cout << endl;
    }
    else {
        if ((t <= sum + total) && (t >= sum + w[k])) {
            s[k] = true;
            SoS(k+1, sum+w[k], total-w[k], w, n, s, t);
            s[k] = false;
            SoS(k+1, sum, total-w[k], w, n, s, t);
        }
    }
}
```

```
void Bai02() {
    // Gia tri vi du, sinh vien co the thay doi de kiem tra
    int w[100] = {3, 5, 6, 7};
    int n = 4;     // So luong so nguyen duong
    int t = 15;    // Tong t can tim tap con

    // Bat dau thuat toan
    bool s[100] = {false};
    int total = 0;

    for (int i = 0; i < n; i++)
        total += w[i];

    if (w[0] <= t && t <= total)
        SoS(0, 0, total, w, n, s, t);
    // Ket thuc thuat toan

    // In ra so phep gan, so phep so sanh
}
//...
```

Sinh viên điều chỉnh mã nguồn được cung cấp sao cho **đếm được số phép gán và số phép so sánh** của hàm `SoS`. Sau đó in ra kết quả tại vị trí được ghi chú trong hàm `Bai02`.

Lưu ý: Việc tăng tham số (ví dụ $k+1$) khi gọi hàm đệ quy KHÔNG cần tính như là phép gán.

3 Câu 3 (3 điểm)

Cho ma trận *mat* có kích thước m_1 dòng, n_1 cột và ma trận *sub_mat* có kích thước m_2 dòng, n_2 cột với $m_2 < m_1; n_2 < n_1$.
Hãy viết hàm kiểm tra xem ma trận *sub_mat* có xuất hiện trong ma trận *mat* không.

- Input: `mat, m1, n1, sub_mat, m2, n2`
- Output: `bool` - Kết quả ma trận `sub_mat` có xuất hiện trong `mat` không. Trả về `true` nếu có xuất hiện và ngược lại trả về `false`.
- Prototype: `bool isIn(int** mat, int m1, int n1, int** sub_mat, int m2, int n2);`
- Sinh viên gọi hàm `isIn` và ghi kết quả trong hàm `Bai03`
- Ví dụ: Cho 2 ma trận `mat` và `sub_mat` như sau:

`mat`, `m1 = 3`, `n1 = 4`

```
5 3 1 2
2 1 9 1
2 8 5 6
```

`sub_mat`, `m2 = 2`, `n2 = 2`:

```
1 2
9 1
```

Gọi hàm `isIn(mat, m1, n1, sub_mat, m2, n2)` ta được kết quả là `true`.

— HẾT —