MÔN : CÔNG NGHỆ JAVA

Bài thực hành 7.2 : Xây dựng chương trình tính tích 2 ma trận dùng multi-thread

I. Mục tiêu :

- Giúp SV làm quen với việc sử dụng môi trường lập trình trực quan NetBeans.
- Giúp SV làm quen với qui trình thiết kế trực quan cửa sổ giao diện chứa nhiều đối tượng giao diện Swing.
- Giúp SV làm quen với việc viết code thực hiện thuật giải song song bằng kỹ thuật multi-thread trên Java.
- Giúp SV thấy rõ tính hiệu quả của thuật giải song song bằng kỹ thuật multi-thread trên Java.

II. Nội dung :

- Dùng NetBeans để thiết kế cửa sổ giao diện của chương trình cho phép người dùng thiết lập số thread cần dùng để tính tích 2 ma trận có kích thước lớn và xem thời gian tính ứng với số thread được dùng.

III. Chuẩn đầu ra :

- Sinh viên nắm vững việc sử dụng môi trường lập trình trực quan NetBeans để thiết kế cửa sổ giao diện của chương trình, định nghĩa hàm xử lý sự kiện cho sự kiện của phần tử giao diện xác định.
- Sinh viên nắm vững việc viết code thực hiện thuật giải song song bằng kỹ thuật multi-thread trên Java.

IV. Qui trình :

1. Chạy NetBean 7.3.1, nếu cửa sổ Project có hiển thị các Project cũ hãy đóng chúng lại.
2. Chọn menu File.New Project để máy hiển thị cửa sổ "New Project", chọn mục "Java" trong Listbox Categories, chọn mục "Java Application" trong Listbox Projects rồi click button Next để hiển thị cửa sổ "New Java Application".
3. Xác định thư mục chứa Project ở textbox "Project Location", nhập "NBThreadDemo1" vào textbox "Project Name", click button Finish để máy tạo thực sự Project. Cửa sổ mã nguồn của class chương trình NBThreadDemo1 hiển thị. Soạn code cho hàm main như sau :

```
public static void main(String[] args) {
    //tạo và hiển thị Form giao diện cho ứng dụng
    MainForm dlg = new MainForm();
    dlg.show();
}
```

4. Trong cửa sổ quản lý Project, làm hiển thị chi tiết package nbthreaddemo1 (hiện đang chứa file miêu tả chương trình NBThreadDemo1.java). Ấn phải chuột trên folder nbthreaddemo1, chọn chức năng New.JFrame Form để máy hiển thị cửa sổ "New JFrame Form", nhập tên Form mới là MainForm, click chuột vào button Finish để máy tạo ra Frame tương ứng, cửa sổ thiết kế Form sẽ hiển thị.
5. Chọn icon Label trong cửa sổ Palette (thường ở góc trên phải màn hình), dời chuột về vị trí trên trái cửa sổ thiết kế và vẽ nó. Quan sát cây quản lý các đối tượng giao diện ở cửa sổ Navigator (thường nằm ở góc dưới trái màn hình), ta thấy Label mới tạo có tên là jLabel1. Ấn phải chuột vào jLabel1 để hiển thị menu lệnh kết hợp, chọn mục "Exit Text" và hiệu chỉnh lại chuỗi hiển thị trên Label là "Nhập số thread cần dùng : ".
6. Chọn icon Text Field trong cửa sổ Palette, dời chuột về vị trí bên phải Label vừa vẽ và vẽ nó. Quan sát cây quản lý các đối tượng giao diện ở cửa sổ Navigator, ta thấy Text Field

mới tạo có tên là jTextField1. Ấn phải chuột vào jTextField1 để hiển thị menu lệnh kết hợp, chọn mục "Exit Text" và hiệu chỉnh lại chuỗi hiển thị trên Text Field là "". Ấn phải chuột vào jTextField1 để hiển thị menu lệnh kết hợp, chọn mục "Change Variable Name" và hiệu chỉnh lại tên nhận dạng cho Text Field là "txtThreads", đây là tên biến để code chương trình truy xuất Text Field này.

7. Chọn icon Button trong cửa sổ Palette, dời chuột về vị trí bên phải TextField vừa vẽ và vẽ nó. Quan sát cây quản lý các đối tượng giao diện ở cửa sổ Navigator, ta thấy Button mới tạo có tên là jButton1. Ấn phải chuột vào jButton1 để hiển thị menu lệnh kết hợp, chọn mục "Exit Text" và hiệu chỉnh lại chuỗi hiển thị trên Button là "Bắt đầu tính". Ấn phải chuột vào jButton1 để hiển thị menu lệnh kết hợp, chọn mục "Change Variable Name" và hiệu chỉnh lại tên nhận dạng cho Button là "btnStart", đây là tên biến để code chương trình truy xuất Button này.
8. Chọn icon ScrollPane trong cửa sổ Palette, dời chuột về vị trí bên dưới các đối tượng đã vẽ và vẽ nó để chiếm hết phần không gian còn lại của Frame.
9. Chọn icon List trong cửa sổ Palette, dời chuột về vị trí bên trong ScrollPane và thả nó ra, máy sẽ vẽ Listbox chiếm hết ScrollPane. Quan sát cây quản lý các đối tượng giao diện ở cửa sổ Navigator, ta thấy Listbox mới tạo có tên là jList1. Ấn phải chuột vào jList1 để hiển thị menu lệnh kết hợp, chọn mục ""Change Variable Name" và hiệu chỉnh lại tên nhận dạng cho Listbox là "lbKetqua", đây là tên biến để code chương trình truy xuất Listbox này.
   Cân chỉnh kích thước và vị trí các đối tượng sao cho có dạng sau :

10. Chọn button "Bắt đầu tính" để hiển thị cửa sổ thuộc tính của nó (thường ở góc dưới phải màn hình). Click button Event trong cửa sổ thuộc tính để máy hiển thị danh sách các sự kiện kết hợp với button. Tìm sự kiện actionPerformed, ấn chuột vào mũi tên chỉ xuống trong listbox bên phải sự kiện và chọn hàm btnStartActionPerformed để máy tạo hàm xử lý sự kiện tương ứng với button. Cửa sổ mã nguồn hiển thị hàm vừa tạo ra với thân rỗng, hãy viết code cho hàm để thực hiện tính tích theo số thread qui định như sau :

```
//hàm tính tích 2 ma trận A, B dùng cnt threads
private void btnStartActionPerformed(java.awt.event.ActionEvent evt) {
    String buf;
    //xác định số thread cần chạy
    cnt = Integer.valueOf(txtThreads.getText().toString());
    t1 = Calendar.getInstance().getTimeInMillis();
    if (cnt == 1)
       { //dùng thuật giải tuần tự
          TinhTich(0, N, 0);
       }
   else //dùng thuật giải song song
```

```
        {
            int i;
            Thread t;
            for (i = 0; i < cnt-1; i++) {
                stateLst[i] = 0;
                //tạo thread mới và truyền các tham số cần thiết cho nó biết
                t = new MyThread(i*N/cnt, (i+1)*N/cnt,i,this);
                t.start();
            }
            //bản thân thread cha sẽ tính N/cnt hàng cuối của ma trận tích
            TinhTich((cnt-1)*N/cnt, N, cnt-1);
        }
    isEnd();
}
```

11. Dời cursor về đầu class MainForm, thêm các lệnh định nghĩa các thuộc tính và các hàm dịch vụ cần dùng như sau :

```
//định nghĩa các biến cần dùng
DefaultListModel lmKetqua;
final int MAXTH=30;
double[][] A;
double[][] B;
double[][] C;
int N;          //số hàng, cột của các ma trận
int[] stateLst = new int [MAXTH];   //danh sách trạng thái hoàn thành các thread
int cnt;        //số thread cần chạy
long t1;        //thời điểm bắt đầu tính tích
//hàm tính các hàng ma trận tích từ sr tới er-1
public void TinhTich (int sr, int er, int id) {
    int h, c, k;
    for (h = sr; h < er; h++)
        for (c = 0; c < N; c++){
            double s = 0;
            for (k = 0; k < N; k++)
                s = s + A[h][k] * B[k][c];
            C[h][c] = s;
        }
    stateLst[id] = 1;
}

//hàm kiểm tra xem các thread đã xong chưa
public synchronized void isEnd() {
        int i;
        //lặp kiểm tra các thread con tính xong
        for (i = 0; i < cnt; i++)
            if (stateLst[i] == 0) return;
        //các thread con đã tính xong
        if (fdisp) return;//nếu đã hiển thị kết quả thì khỏi làm nữa
        fdisp = true;
        //tính và hiển thị thời gian chạy
        long t2 = Calendar.getInstance().getTimeInMillis();
```

```
        long diff = t2 - t1;
        long diffS = (diff / 1000)%60;
        long diffM = diff / (60 * 1000);
        diff = diff % 1000;
String buf = cnt+" threads ==> Thời gian chạy là "+ diffM+ " phút "+ diffS+ " giây "+diff+ " ms";
        lmKetqua.addElement(buf);
    }
```

12. Dời cursor về cuối hàm constructor của MainForm, thêm các lệnh xóa nội dung ban đầu của các đối tượng giao diện như sau :

```
    //hàm khởi tạo Frame
    public MainForm() {
        initComponents();
        //thiết lập nội dung ban đầu null cho Textbox và Listbox
        txtThreads.setText("");
        lmKetqua = new DefaultListModel();
        lbKetqua.setModel((ListModel) lmKetqua);

        //khởi tạo các ma trận A, B theo số liệu giả
        N = 1000;
        A = new double[N][N];
        B = new double[N][N];
        C = new double[N][N];
        int h, c;
        for (h = 0 ; h < N; h++)
            for (c = 0; c < N; c++)
                A[h][c] = B[h][c] = c;
    }
```

13. Dời cursor về đầu file mã nguồn, ngay sau lệnh package, thêm các import các packages cần dùng như sau :

```
    //import các packages cần dùng
    import java.util.Calendar;
    import javax.swing.*;
```

14. Về cửa sổ quản lý Project, ấn phải chuột trên folder nbthreaddemo1, chọn chức năng New.Java Class để máy hiển thị cửa sổ "New Java Class", nhập tên class mới là MyThread, click chuột vào button Finish để máy tạo ra class tương ứng, cửa sổ soạn mã nguồn cho class sẽ hiển thị. Hãy viết code cho class này như sau :

```
    //class quản lý thread con
    public class MyThread extends Thread {
    //định nghĩa các thuộc tính cần sử dụng
    int sr, er, id;
    MainForm frm;
    //định nghĩa tác vụ khởi tạo nhận tham số
    MyThread(int sr, int er, int id, MainForm frm) {
        super("Demo Thread");
        this.sr = sr; this.er = er; this.id = id;
        this.frm = frm;
    }

    //định nghĩa tác vụ mà thread sẽ chạy
```

```
public void run() {
    frm.TinhTich(sr,er,id);          //tính các hàng ma trận tích từ sr tới er-1
    frm.stateLst[id] = 1;   //ghi nhận trạng thái hoàn thành
    frm.isEnd();            //gởi thông báo về thread cha
}
```

15. Chọn menu Run.Run Project để dịch và chạy thử chương trình. Nếu có lỗi từ vựng và cú pháp thì sữa, nếu có lỗi run-time thì debug (thông qua các chức năng trong menu Debug) để xác định lỗi rồi sữa lỗi.
16. Nếu chương trình hết lỗi, cửa sổ chương trình sẽ hiển thị.
17. Hãy nhập số thread cần chạy rồi click button "Bắt đầu tính" và chờ xem kết quả thời gian chạy.
18. Lặp lại bước 17 nhiều lần với số thread khác nhau (<30) để kiểm tra kết quả thời gian chạy cho từng trường hợp.