

LAB 2

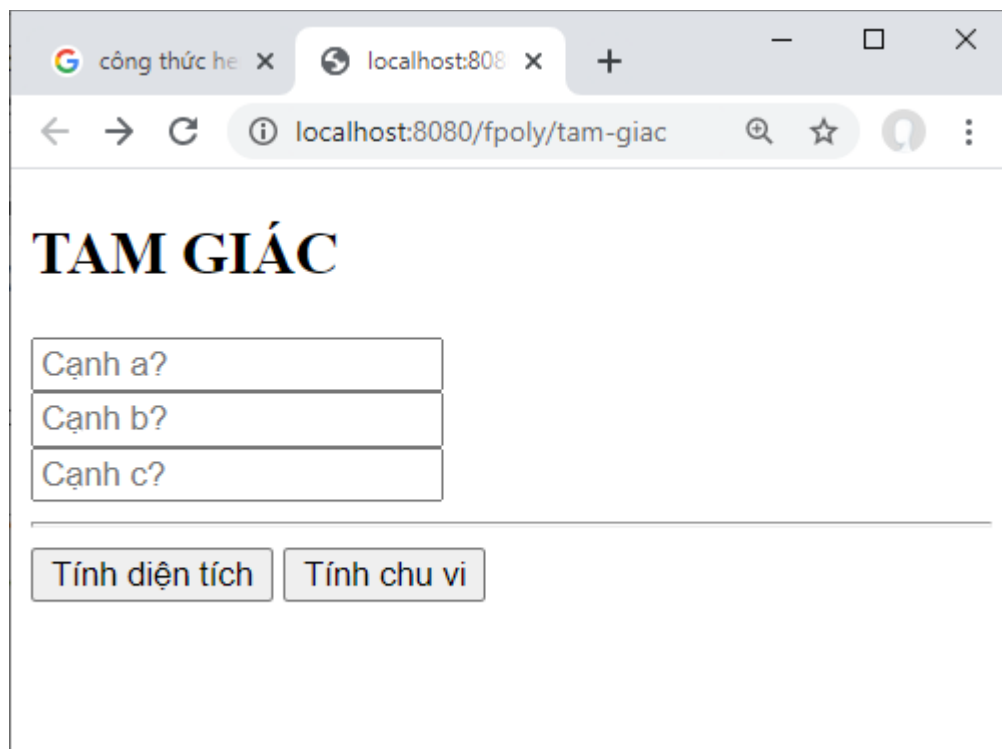
MỤC TIÊU:

- Tổ chức được servlet
- Phân biệt được doPost() và doGet()
- Nhận và xử lý được tham số form phức tạp
- Giải thích được cơ chế xử lý đa luồng của công nghệ Servlet/JSP

PHẦN I: TỔ CHỨC SERVLET VÀ XỬ LÝ FORM ĐƠN GIẢN

BÀI 1: XỬ LÝ FORM NHIỀU NÚT CÓ PHÂN BIỆT POST VÀ GET

Xây dựng trang web cho phép tính diện tích và chu vi của một tam giác như mô tả các hình sau:



The screenshot shows a web browser window with two tabs: 'công thức he' and 'localhost:808'. The address bar shows 'localhost:8080/fpoly/tam-giac'. The page content includes the title 'TAM GIÁC' in a large, bold, serif font. Below the title are three input fields labeled 'Cạnh a?', 'Cạnh b?', and 'Cạnh c?'. At the bottom of the form are two buttons: 'Tính diện tích' (Calculate area) and 'Tính chu vi' (Calculate perimeter).

Hình 1: Mới chạy



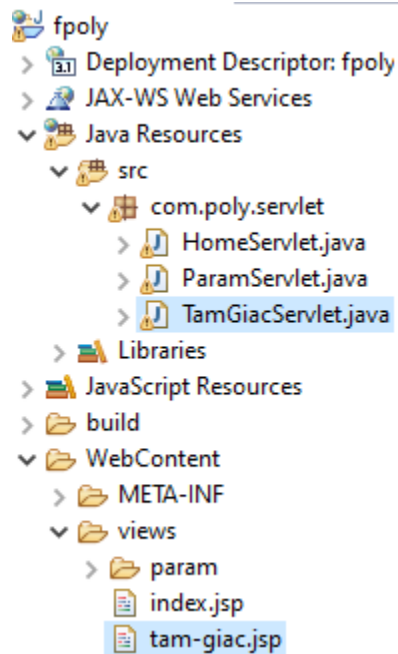
Hình 2: $a=1$, $b=2$ và $c=3 \Rightarrow$ [Tính chu vi].Click



Hình 3: $a=2$, $b=3$ và $c=4 \Rightarrow$ [Tính chu vi].Click

Hướng dẫn:

- Tạo TamGiacServlet.java và tam-giac.jsp



- Viết mã cho tam-giac.jsp

```
<form action="/fpoly/tam-giac" method="post">
    <input name="a" placeholder="Cạnh a?"> <br>
    <input name="b" placeholder="Cạnh b?"> <br>
    <input name="c" placeholder="Cạnh c?"> <hr>
    <button formaction="/fpoly/dien-tich">Tính diện tích</button>
    <button formaction="/fpoly/chu-vi">Tính chu vi</button>
</form>
<h3>${message}</h3>
```

- Tổ chức mã TamGiacServlet.java

```
@WebServlet({" /tam-giac" "/dien-tich" "/chu-vi"})
public class TamGiacServlet extends HttpServlet{
    @Override
    protected void doGet(HttpServletRequest req, HttpServletResponse resp)
        throws ServletException, IOException {
        req.getRequestDispatcher("/views/tam-giac.jsp").forward(req, resp);
    }
    @Override
```

```

protected void doPost(HttpServletRequest req, HttpServletResponse resp)
    throws ServletException, IOException {
    double a = Double.parseDouble(req.getParameter("a"));
    // đọc b và c ở đây
    if( (a + b > c) && (a + c > b) && (b + c > a) ) {
        // thực hiện tính diện tích hoặc chu vi ở đây
    }
    else {
        req.setAttribute("message",
            "Không thỏa mãn các cạnh của một tam giác!");
    }
    req.getRequestDispatcher("/views/tam-giac.jsp").forward(req, resp);
}
}

```

- Viết mã tính diện tích hoặc chu vi tùy vào lựa chọn của người dùng

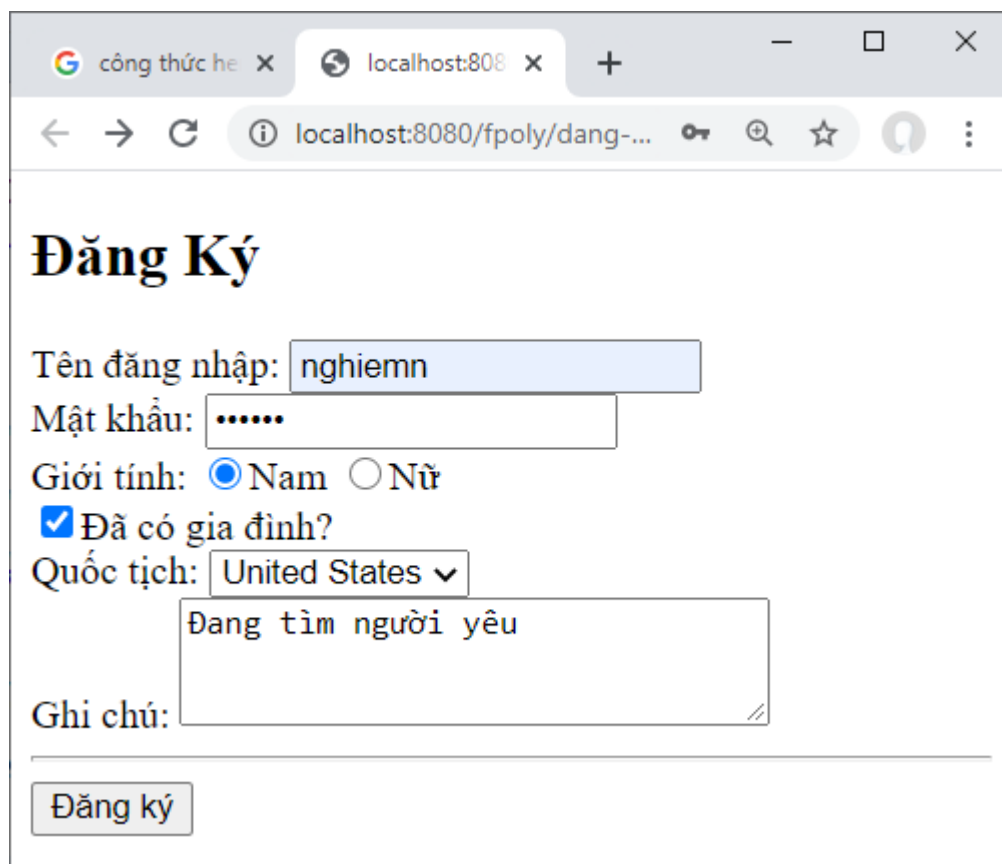
```

double chuVi = (a + b + c);
String uri = req.getRequestURI();
if(uri.contains("dien-tich")) { // [Tính diện tích].Click
    double dienTich = Math.sqrt(chuVi * (a + b - c) * (a + c - b) * (b + c - a))/4;
    req.setAttribute("message", "Diện tích của tam giác là " + dienTich);
}
else{ // [Tính chu vi].Click
    req.setAttribute("message", "Chu vi của tam giác là " + chuVi);
}
}

```

BÀI 2: XỬ LÝ FORM VỚI ĐA DẠNG PHẦN TỬ

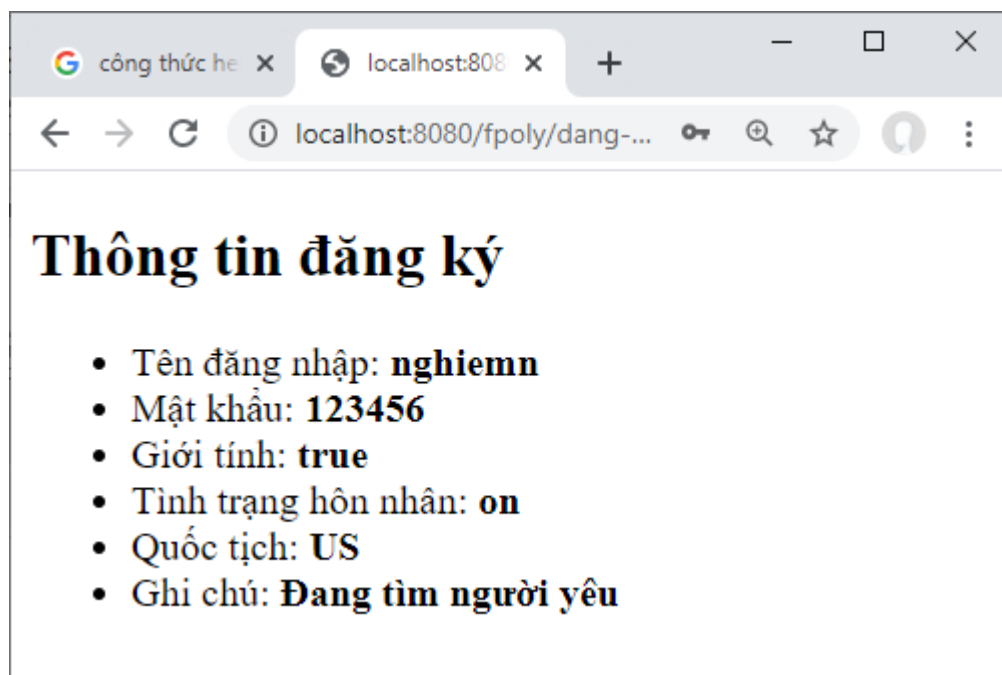
Xây dựng trang web cho phép đọc dữ liệu form đăng ký và xuất ra màn hình Console như hình sau:



The screenshot shows a web browser window with the address bar displaying 'localhost:8080/fpoly/dang-...'. The page title is 'Đăng Ký'. The form contains the following fields and options:

- Tên đăng nhập:
- Mật khẩu:
- Giới tính: ☒ Nam ☐ Nữ
- ☒ Đã có gia đình?
- Quốc tịch:
-
- Ghi chú:
-

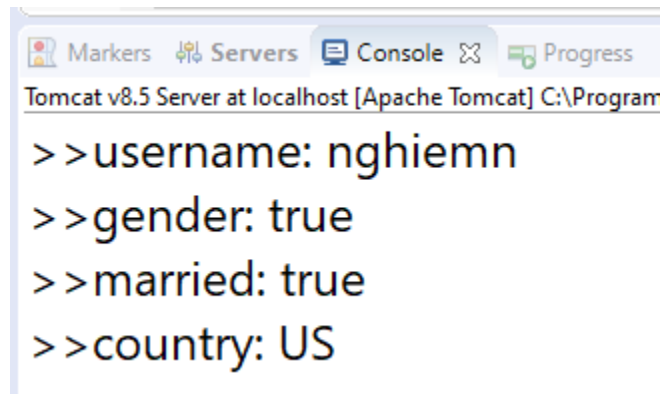
Hình 1: Form đăng ký



The screenshot shows a web browser window with the address bar displaying 'localhost:8080/fpoly/dang-...'. The page title is 'Thông tin đăng ký'. The form displays the following information:

- Tên đăng nhập: **nghiemn**
- Mật khẩu: **123456**
- Giới tính: **true**
- Tình trạng hôn nhân: **on**
- Quốc tịch: **US**
- Ghi chú: **Đang tìm người yêu**

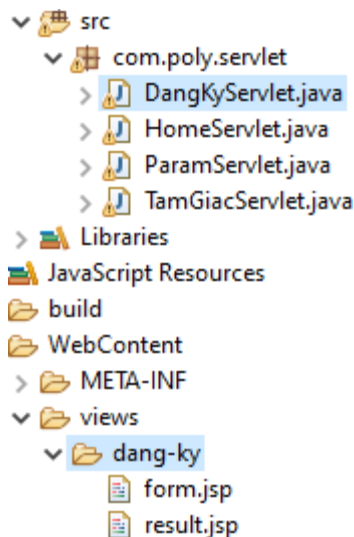
Hình 2: Kết quả hiển thị



Hình 3: Đọc tham số trong Servlet

Hướng dẫn

- Tạo DangKyServlet.java, form.jsp và result.jsp như hình sau



- Viết mã cho DangKyServlet.java
doGet() => form.jsp
doPost() => result.jsp
- Viết mã cho form.jsp

```
<form action="/fpoly/dang-ky" method="post">  
  Tên đăng nhập: <input name="username"><br>  
  Mật khẩu: <input name="password" type="password"><br>  
  Giới tính:  
  <input name="gender" type="radio" value="true"> Nam  
  <input name="gender" type="radio" value="false"> Nữ<br>
```

```

☐ Đã có gia đình? <br>
Quốc tịch: <select name="country">
    <option value="VN">Việt Nam</option>
    <option value="US">United States</option>
</select> <br>
Ghi chú: <textarea name="notes" rows="3" cols="30"> </textarea>
<hr>
<button>Đăng ký</button>
</form>

```

- Viết mã cho result.jsp để hiển thị các tham số của form nhập

```

<ul>
    <li>Tên đăng nhập: <b>${param.username}</b> </li>
    <li>Mật khẩu: <b>${param.password}</b> </li>
    <li>Giới tính: <b>${param.gender}</b> </li>
    <li>Tình trạng hôn nhân: <b>${param.married}</b> </li>
    <li>Quốc tịch: <b>${param.country}</b> </li>
    <li>Ghi chú: <b>${param.notes}</b> </li>
</ul>

```

- Bổ sung mã cho DangKyServlet.java để đọc các tham số form trong servlet (Hãy viết đoạn mã sau vào vị trí trước khi forward sang result.jsp)

```

req.setCharacterEncoding("utf-8");
resp.setCharacterEncoding("utf-8");

String username = req.getParameter("username");
boolean gender = Boolean.parseBoolean(req.getParameter("gender"));
boolean married = (req.getParameter("married") != null);
String country = req.getParameter("country");

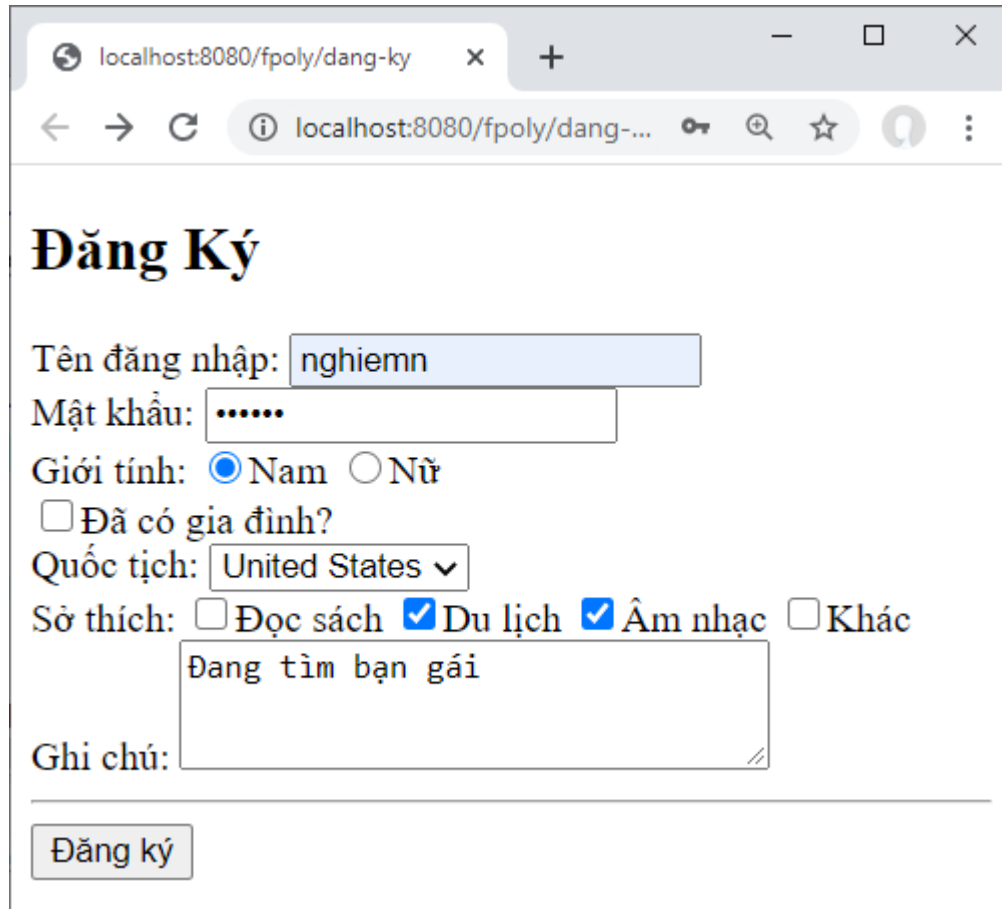
System.out.println(">>username: " + username);
System.out.println(">>gender: " + gender);
System.out.println(">>married: " + married);
System.out.println(">>country: " + country);

```

PHẦN II: XỬ LÝ THAM SỐ PHỨC TẠP VÀ TÌM HIỂU ĐA LƯỜNG

BÀI 3: LÀM VIỆC VỚI THAM SỐ NHIỀU GIÁ TRỊ

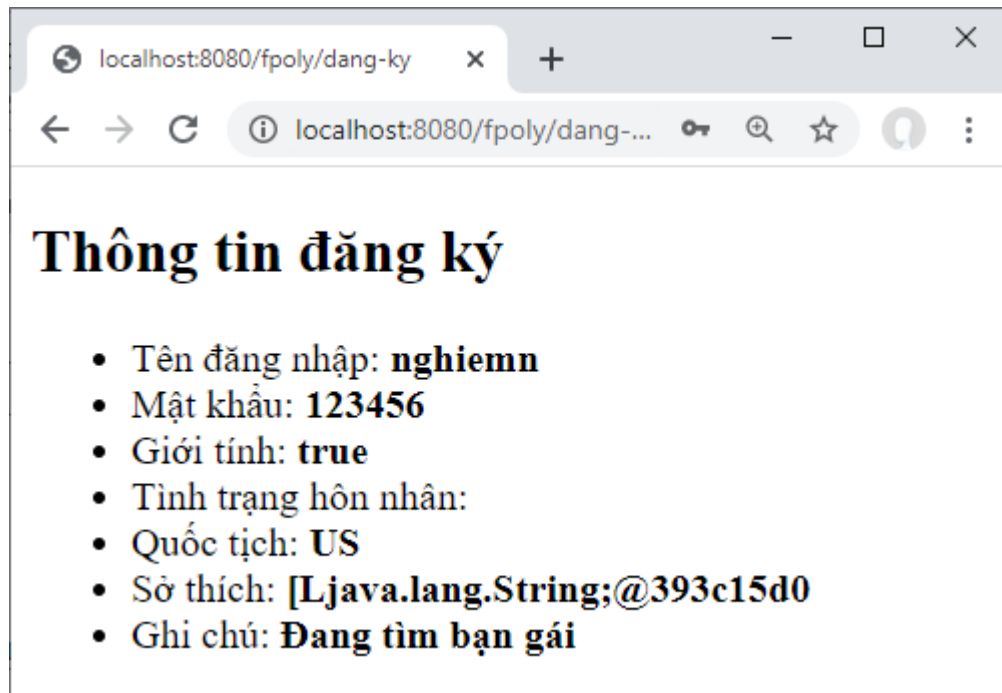
Hãy thêm sở thích vào form đăng ký, sau đó lập trình để đọc được các giá trị được chọn cũng như hiển thị lên giao diện.



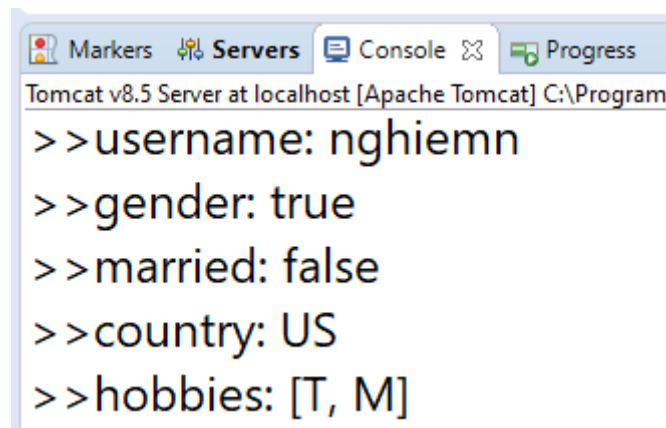
The screenshot shows a web browser window with the address bar displaying 'localhost:8080/fpoly/dang-ky'. The page title is 'Đăng Ký'. The form contains the following fields and options:

- Tên đăng nhập:
- Mật khẩu:
- Giới tính: ☒ Nam ☐ Nữ
- ☐ Đã có gia đình?
- Quốc tịch:
- Sở thích: ☐ Đọc sách ☒ Du lịch ☒ Âm nhạc ☐ Khác
-
- Ghi chú:
-

Hình 1: Thêm sở thích



Hình 2: Hiển thị sở thích trên JSP



Hình 3: Đọc các sở thích được chọn trong Servlet

Hướng dẫn:

- Thêm sở thích vào form.jsp

Sở thích:

```
<input name="hobbies" type="checkbox" value="R">Đọc sách
<input name="hobbies" type="checkbox" value="T">Du lịch
<input name="hobbies" type="checkbox" value="M">Âm nhạc
<input name="hobbies" type="checkbox" value="O">Khác<br>
```

- Hiển thị sở thích trên result.jsp

```
<li>Sở thích: <b>${paramValues.hobbies}</b></li>
```

- Lập trình đọc sở thích trong servlet

```
String[] hobbies = req.getParameterValues("hobbies");
System.out.println(">>hobbies: " + Arrays.toString(hobbies));
```

BÀI 4: VÒNG ĐỜI SERVLET

Hãy xây dựng bộ đếm số lần truy cập một servlet.

Hướng dẫn:

- Tạo một HitCounterServlet và override 3 phương thức
 - init()
 - service()
 - destroy()
- Khai báo 2 field sau đây vào HitCounterServlet
 - int count;
 - Path path = Paths.get("c:/temp/count.txt");
- Trong init() hãy đọc số đếm từ file và khởi đầu cho count
 - count = Integer.parseInt(Files.readAllLines(path).get(0));
- Trong service() tăng count lên một và chia sẻ biến count vào request trước khi chuyển sang hit-counter.jsp
 - count++;
 - req.setAttribute("count", count);
- Trong hit-counter.jsp hiển thị giá trị của count đã được chia sẻ từ HitCounterServlet
 - \${count}
- Trong destroy() lưu số đếm vào file
 - Files.write(path, String.valueOf(count).getBytes(), StandardOpenOption.WRITE);
- **Chú ý: cần tạo file count.txt trong thư mục temp tại ổ c: và nhập vào đó một số nào đó, ví dụ: 2020.**