TRƯỜNG ĐẠI HỌC KIẾN TRÚC ĐÀ NẴNG

# KHOA CÔNG NGHỆ THÔNG TIN

**BÁO CÁO ĐỒ ÁN GIẢI THUẬT VÀ LẬP TRÌNH**



# ĐỀ TÀI:

**Tìm hiểu bài toán tìm kiếm nhị phân và xây dựng ví dụ minh hoạ**.  
  
**Giảng viên hướng dẫn :** ĐỖ PHÚC HẢO

**Sinh viên thực hiện :** Nguyễn Duy Tân  
 Võ Hà An  
 **Lớp:** 19CT2

[KHOA CÔNG NGHỆ THÔNG TIN 1](#_Toc75208939)

[ĐỀ TÀI: Tìm hiểu thuật toán tìm kiếm nhị phân và xây dựng ví dụ minh họa 1](#_Toc75208940)

[1 Thuât toán BINARY SEARCH ( Tìm kiếm nhị phân) 2](#_Toc75208941)

[2.1 Ý tưởng và mô tả thuật toán tìm kiếm nhị phân 3](#_Toc75208941)

[2.2 Đánh giá độ phức tạp của thuật toán 3](#_Toc75208941)

[3. Triển khai thuật toán Binary Search trong Java: 4](#_Toc75208942)

[Bước 1. Xây dựng thuật toán binary search trong java 4](#_Toc75208941)

[Bước 2. Xây dựng thuật toán sắp xếp phân đoạn ( Quicksort) 4](#_Toc75208941)

[Bước 3. Thiết lập hàm random 1 mảng ngẫu nhiên không lặp 5](#_Toc75208941)

[Bước 4. Triển khai thuật toản trong hàm main 5](#_Toc75208941)

[4. Triển khai thuật toán Binary Search trong Javaswing 6](#_Toc75208941)

[Bước 1. Xây dựng các thuộc tính đối tượng trong java swing 6](#_Toc75208941)

[Bước 2. Xây dựng giao diện Binary Search trong java swing 6](#_Toc75208941)

[Bước 3 : Bắt sự kiện cho giao diện 6](#_Toc75208941)

[Bước 3.1 : Tạo mảng , sắp xếp , tìm kiếm 6](#_Toc75208941)

[Bước 3.2 : Tìm min , max và thoát 8](#_Toc75208941)

[Bước 4 : Dùng hàm binary search 8](#_Toc75208941)

[Bước 5 : Dùng hàm Quicksort 9](#_Toc75208941)

[5 : Thuật toán game đoán số 10](#_Toc75208941)

[6 : Xây dựng thuật toán game đoán số trong java 11](#_Toc75208941)

[7 : Xây dựng thuật toán game đoán số trong java swing 12](#_Toc75208941)

[Bước 1 : xây dựng các thuộc tính đối tượng trong java 12](#_Toc75208941)

[Bước 2 : Xây dựng giao diện Game đoán số trong java 12](#_Toc75208941)

[Bước 3 : Bắt sự kiện 12](#_Toc75208941)

[8 : Tổng kết 15](#_Toc75208941)

**1. Thuật toán BINARY SEARCH(Tìm kiếm nhị phân):  
Thuật toán Binary Serach** (**Tìm kiếm nhị phân**) là một thuật toán tìm kiếm tuyến tính cao cấp hơn với thời gian chạy là O(logN).   
  
Đối với các danh sách lớn, thuật toán này tốt hơn hẳn tìm kiếm tuyến tính, nhưng nó đòi hỏi **danh sách phải được sắp xếp từ trước** và đòi hỏi khả năng truy nhập ngẫu nhiên (random access).  
  
Thuật toán tìm kiếm nhị phân thường dùng để tìm kiếm phần tử trong một danh sách đã được sắp xếp, ví dụ như trong một danh bạ điện thoại sắp xếp theo tên, có thể tìm kiếm số điện thoại của một người theo tên người đó.  
  
Thuật toán tìm kiếm nhị phân chạy nhanh hơn tìm kiếm tuần tự nhưng cũng có một số nhược điểm. Nó chậm hơn bảng băm.  
  
Nếu nội dung danh sách bị thay đổi thì danh sách phải được sắp xếp lại trước khi sử dụng tìm kiếm nhị phân.  
  
Mà thao tác này thường tốn nhiều thời gian.  
  
**2.1 Ý tưởng và Mô tả thuật toán Binary Search:**Vì thuật toán **Binary Search** yêu cầu mảng đã được sắp xếp. Thế nên, đầu vào của chúng ta là một mảng đã được sắp xếp.  
  
Do đó, thuật toán tìm kiếm nhị phân sẽ so sánh giá trị đích với phần tử ở giữa của mảng (mảng được chia mảng ra làm 2 phần bên trái và bên phải phần tử đó)  
  
Nếu chúng không bằng nhau thì dĩ nhiên một nửa không chứa mục tiêu sẽ bị bỏ đi.  
  
Và việc tìm kiếm tiếp tục ở nửa còn lại, một lần nữa lấy phần tử ở giữa được chọn để so sánh với giá trị đích và lặp lại điều này cho đến khi tìm thấy giá trị đích.  
  
Nếu tìm kiếm kết thúc với nửa còn lại trống, mục tiêu không nằm trong mảng.  
  
Mặc dù ý tưởng rất đơn giản, nhưng việc thực hiện tìm kiếm nhị phân chính xác đòi hỏi phải chú ý đến một số điểm tinh tế về điều kiện thoát và tính toán điểm giữa của nó.  
  
Về cơ bản, các bước thực hiện tìm kiếm nhị x trong mảng như sau:

* So sánh x với phần tử ở giữa
* Nếu x khớp với phần tử ở giữa, chúng ta trả về chỉ số giữa
* Nếu x lớn hơn phần tử giữa, thì x chỉ có thể nằm trong nửa phân đoạn bên phải sau phần tử mid. Vì vậy, chúng ta chỉ tìm kiếm ở nữa phải của mảng
* Nếu x nhỏ hơn phần tử giữa tiếp tục tìm ở nửa bên trái
* Lặp lại đến khi tìm ra x hoặc trả về null khi x không nằm trong mảng
* Ví dụ ta có mảng: A[2, 4, 9, 10, 11, 22, 24, 31, 48, 56, 76, 86,90]  
  Nhiệm vụ: Tìm vị trí của số 10 trong mảng

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 2 | 4 | 9 | 10 | 11 | 22 | 24 | 31 | 48 | 56 | 76 | 86 | 90 |

0 1 2 3 4 5 6 7 8 9 10 11 12

* Đầu tiên, ta so sánh số 10 với phần tử ở giữa thì thấy 10 < 22. Bỏ phần bên phải.

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 2 | 4 | 9 | 10 | 11 | 22 | 24 | 31 | 48 | 56 | 76 | 86 | 90 |

0 1 2 3 4 5 6 7 8 9 10 11 12  
 **Chặn dưới giữa=(dưới + trên)/2** chặn trên.

* Tiếp tục với phần bên trái với phần tử giữa là 9. Ta có 10 > 9. Vì thế ta bỏ phần bên trái.

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 2 | 4 | 9 | 10 | 11 | **22** | **24** | **31** | **48** | **56** | **76** | **86** | **90** |

0 1 2 3 4 5 6 7 8 9 10 11 12

Chặn dưới giữa Chặn trên

\*Tiếp tục với phần bên phải. So sánh phần tử ở giữa (với Giữa = (Chặn dưới + Chặn trên) / 2). Ta tìm thấy giá trị 10 ở vị trí 3.

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 2 | 4 | 9 | 10 | 11 | 22 | 24 | 31 | 48 | 56 | 76 | 86 | **90** |

0 1 2 3 4 5 6 7 8 9 10 11

## 2.2 Đánh giá độ phức tạp của thuật toán

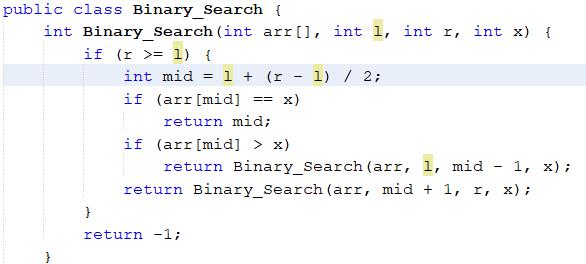
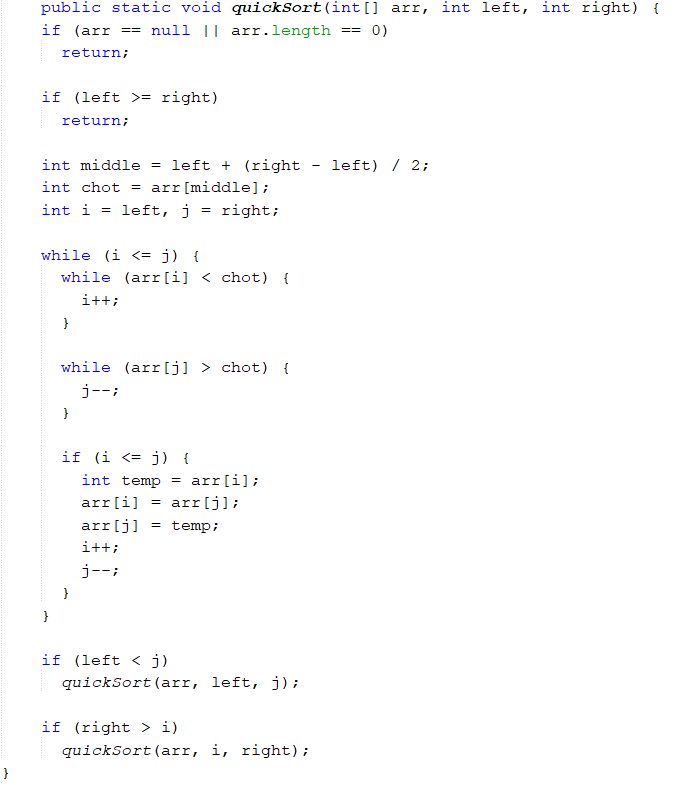
Thuật toán cần ít nhất {\displaystyle \left\lceil \log \_{2}(n)\right\rceil }[Log2(2)] bit để mã hóa một con trỏ tới một phần tử trong mảng có {\displaystyle n}n phần tử.Do đó, độ phức tạp không gian của tìm kiếm nhị phân là O(log n){\displaystyle O(\log n)}. Ngoài ra, thuật toán còn cần {\displaystyle O(n)}O(n) không gian để lưu trữ mảng.

**Hiệu suất trường hợp tốt nhất:**[O(n)](https://vi.wikipedia.org/wiki/K%C3%AD_hi%E1%BB%87u_O_l%E1%BB%9Bn#Orders_of_common_functions)

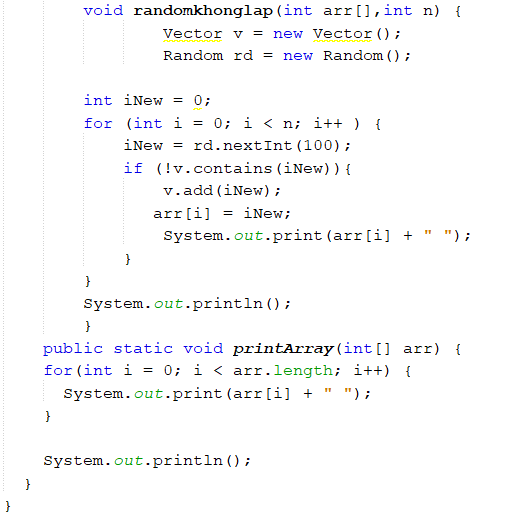
**Hiệu suất trường hợp tệ nhất:**[O(log n)](https://vi.wikipedia.org/wiki/K%C3%AD_hi%E1%BB%87u_O_l%E1%BB%9Bn#Orders_of_common_functions)

## 3. Triển khai thuật toán Binary Search trong Java:

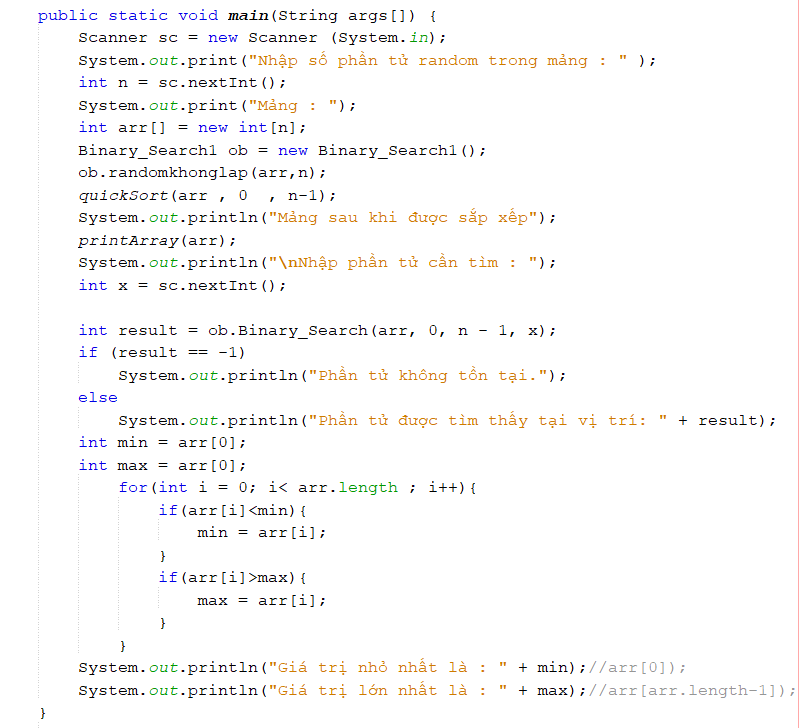
**Bước 1 : xây dựng thuật toán Binary Search trong java**

**Bước 2 : Xây dựng thuật toán sắp xếp phân đoạn (QuickSort)** 

**Bước 3 : Thiết lập hàm random 1 mảng ngẫu nhiên không lặp**



**Bước 4 : Triển khai thuật toán trong hàm Main**

****

Kết quả khi chạy ra là :

Nhập số phần tử random trong mảng : 10

Mảng : 89 10 48 80 56 94 88 91 72 59

Mảng sau khi được sắp xếp

10 48 56 59 72 80 88 89 91 94

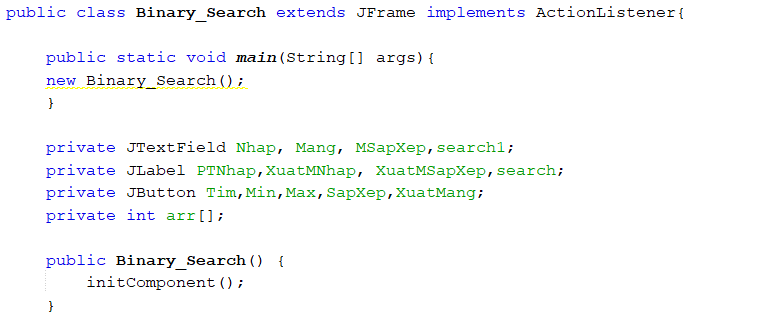
Nhập phần tử cần tìm : 56

Phần tử được tìm thấy tại vị trí: 2

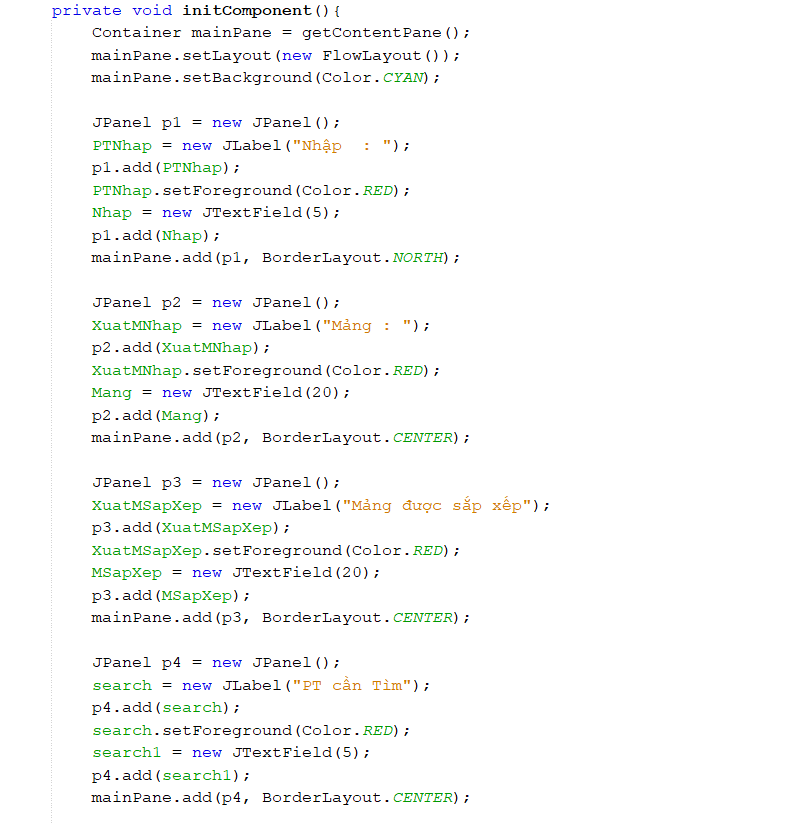
Giá trị nhỏ nhất là : 10

Giá trị lớn nhất là : 94

**4.** **Triển khai thuật toán Binary Search trong Javaswing:  
Bước 1 : xây dựng các thuộc tính đối tượng trong java swing**



**Bước 2 : Xây dựng giao diện Binary Search trong java swing**

****

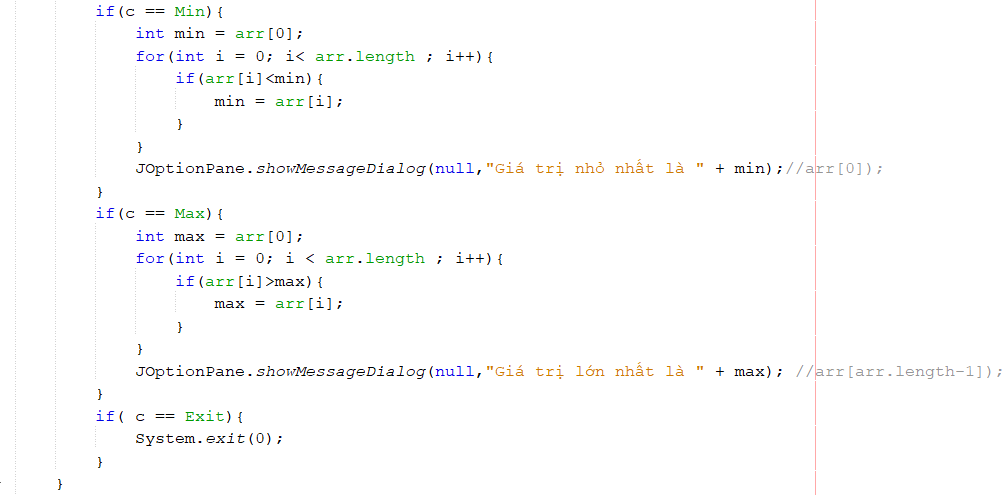
****

**Bước 3 : Bắt sự kiện cho giao diện**

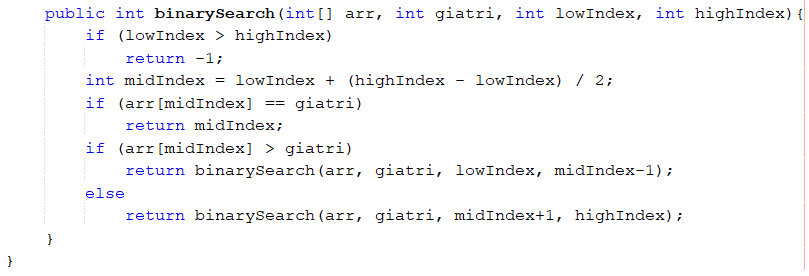
**3.1 : Tạo mảng , sắp xếp , tìm kiếm**

****

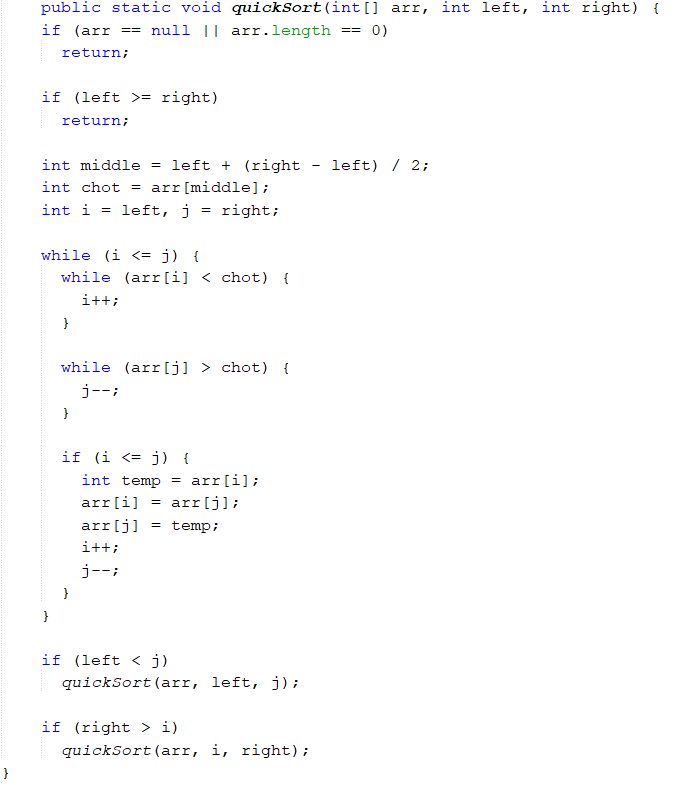
**3.1 : Tìm min , max và thoát**

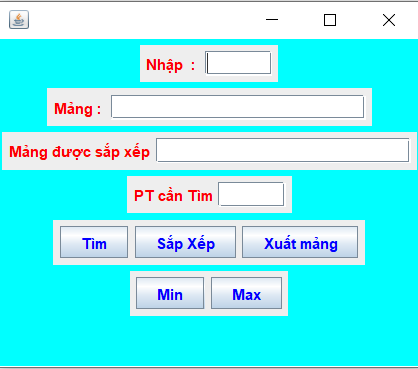
****

**Bước 4 : Dùng hàm Binary Search**

****

**Bước 5 : Dùng hàm Quicksort**



**Khi chạy chương trình ta có kết quả:**  
  
**5. Thuật toán game đoán số:**

**Game: Đoán số (Guess It)**

● Game đoán số là game rất phổ biến và áp dụng triệt để thuật toán tìm kiếm nhị phân trong một mảng số nguyên đã sắp xếp sẵn.

● Dựa trên nguyên tắc có thứ tự của một mảng các số tự nhiên.

● Game thể hiện sự nhanh trí , quá trình suy đoán , lập luận của người chơi.

● Game đoán số dựa vào sự phân vùng dữ liệu để đoán giá trị (đoán số) nằm trong vùng nào để thu hẹp lại không gian tìm kiếm và xử lý dữ liệu.

**Luật chơi**

● Rất đơn giản người chơi chỉ cần nghĩ một con số ngẫu nhiên,

Sau đó bạn hay trả lời những câu hỏi mà máy tính đặt ra

○ Nếu đúng từ bàn phím gỡ chữ “y”

○ Nếu sai từ bàn phím gỡ chữ “n”

**Mục tiêu**

● Mục tiêu của nhóm là thiết kế một game đơn giản có thể chạy tốt trên máy tính. Và game “Đoán số” là sự lựa chọn của nhóm.

● Game “Đoán số” dựa trên luật chơi đơn giản , dễ hiểu đáp ứng nhu cầu về trình độ và phù hợp với lứa tuổi.

**Mô tả thành các bước (bằng lời)**

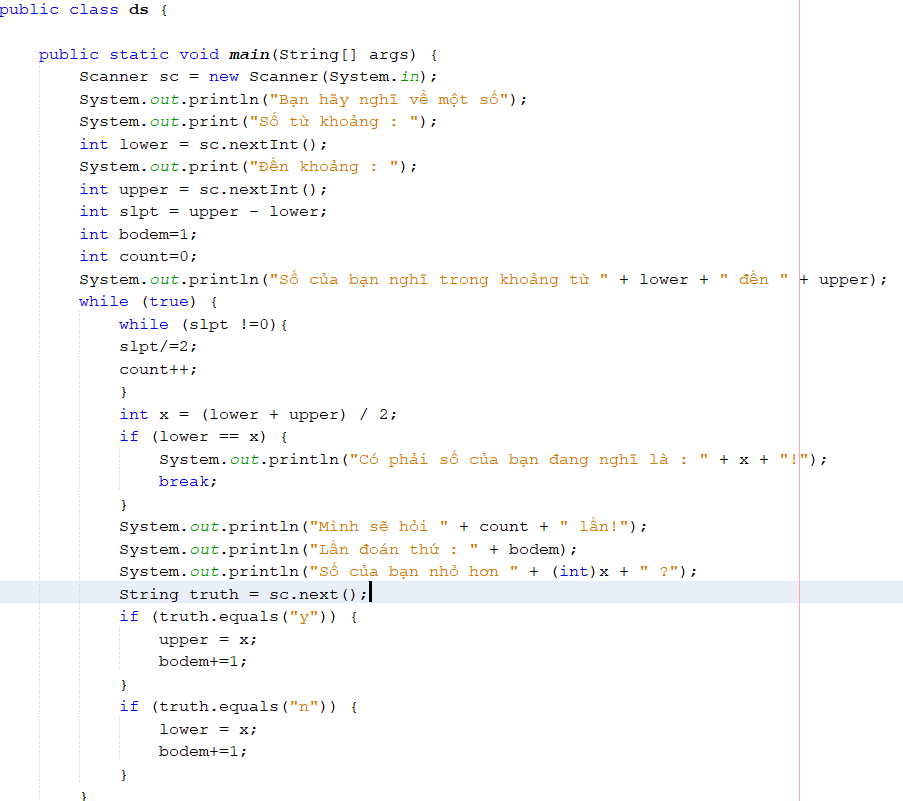
● Người chơi nghĩ số

● Nhập con số từ khoảng … đến khoảng … để máy đoán

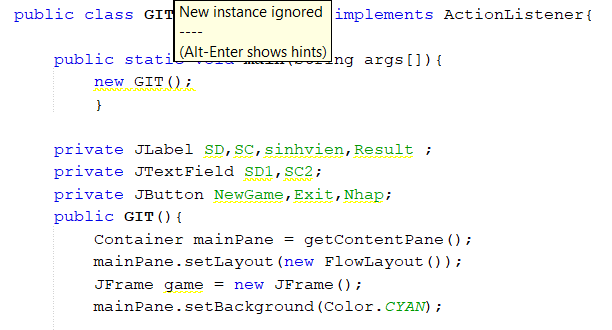
● Người chọn câu trả lời phù hợp ( yes of no )

● Lặp lại đến khi chỉ còn 1 số

**6. Xây dựng thuật toán game đoán số trong java :**

****

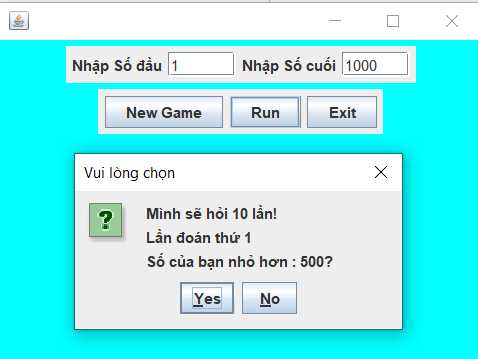
**7. Xây dựng giao diện game đoán số trong java swing :**

**Bước 1 : xây dựng các thuộc tính đối tượng trong java** 

**Bước 2 : Xây dựng giao diện Game đoán số trong java**

**Bước 3 : Bắt sự kiện**

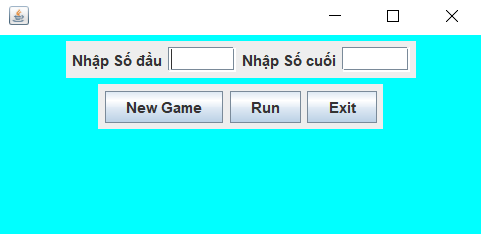
****

**Khi chạy chương trình ta có kết quả:**

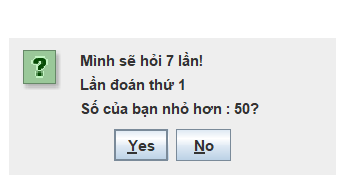
**Chơi thử nghiệm**

● Ví dụ số mà bạn nghĩ cụ thể là số “15” trong khoảng từ “1” đến “100”, chúng ta bắt đầu “game đoán số”

● Giao diện bắt đầu của game đoán số



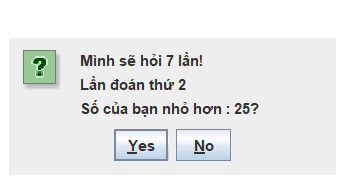
● Lần đoán thứ 1 :



Vì số của mình đang nghĩ là số “15” nhỏ hơn “50” nên khi máy hỏi

“Số của bạn nhỏ hơn : 50?” thì chúng ta click vào nút Yes.

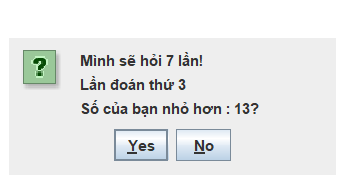
● Lần đoán thứ 2 :



Vì số của mình đang nghĩ là số “15” nhỏ hơn “25” nên khi máy hỏi

“Số của bạn nhỏ hơn : 25?” thì chúng ta click vào nút Yes.

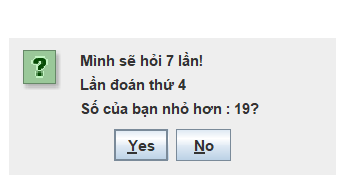
● Lần đoán thứ 3 :



Vì số của mình đang nghĩ là số “15” lớn hơn “13” nên khi máy hỏi

“Số của bạn nhỏ hơn : 13?” thì chúng ta click vào nút No.

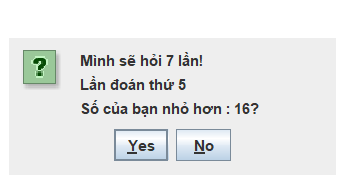
● Lần đoán thứ 4 :



Vì số của mình đang nghĩ là số “15” nhỏ hơn “19” nên khi máy hỏi

“Số của bạn nhỏ hơn : 19?” thì chúng ta click vào nút Yes.

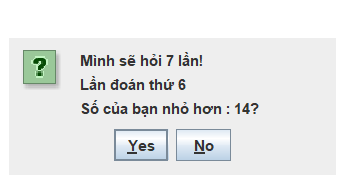
● Lần đoán thứ 5 :



Vì số của mình đang nghĩ là số “15” nhỏ hơn “16” nên khi máy hỏi

“Số của bạn nhỏ hơn : 16?” thì chúng ta click vào nút Yes.

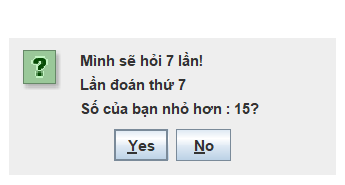
● Lần đoán thứ 6 :



Vì số của mình đang nghĩ là số “15” lớn hơn “14” nên khi máy hỏi

“Số của bạn nhỏ hơn : 14?” thì chúng ta click vào nút No.

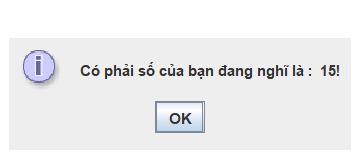
● Lần đoán thứ 7 :



Vì số của mình đang nghĩ là số “15” nhưng câu hỏi của máy hỏi là

“Số của bạn nhỏ hơn : 15?” nên chúng ta click vào nút No.

● Sau 7 lần đoán thì máy đã đoán đúng số của mình đang nghĩ và dừng game



**8. Tổng kết**

Trên đây là toàn bộ chương trình xây dựng “giao diện tìm kiếm nhị phân” và “ Game đoán số”, qua nghiên cứu và kiến thức học được cùng với sự hướng dẫn của giáo viên bộ môn, sự giúp đỡ của các bạn và các thành viên trong nhóm hoàn thiện bài tập này. Qua đó, nhằm giúp cho thế giới game ngày càng đa dạng và phong phú. Nhóm chỉ dừng lại mức độ đơn giản , nhằm giúp cho các bạn có cái nhìn tổng quát về thuật toán tìm kiếm nhị phân.

Với trình độ hiểu biết : Khả năng tìm kiếm , đọc hiểu tài liệu: Kỹ năng lập trình còn hạn chế nên không tránh khỏi những sai sót rất mong nhận được sự thông cảm và đóng góp ý kiến của thầy và các bạn để kỹ năng lập trình cũng như kinh nghiệm của chúng em ngày càng mở rộng và nâng cao hơn. Nếu có cơ hội tiếp theo, hi vọng rằng phiên bản mới của game sẽ có thêm các đặc tính mới hơn

Một lần nữa nhóm chúng em xin chân thành cảm ơn sự giúp đỡ, hướng dẫn, chỉ bảo tận tình của thầy và các bạn để chúng tôi hoàn thành được chương trình này.