**ĐẠI HỌC QUỐC GIA THÀNH PHỐ HỒ CHÍ MINH**

**TRƯỜNG ĐẠI HỌC KHOA HỌC TỰ NHIÊN**

**KHOA CÔNG NGHỆ THÔNG TIN**



**BÁO CÁO ĐỒ ÁN**

**LẬP TRÌNH GAME HANG-MAN**

**BẰNG HỢP NGỮ MIPS**

**Giảng viên hướng dẫn: Lê Viết Long­­**

**Lớp: 18\_1**

Thành phố Hồ Chí Minh, tháng 6 năm 2020

Mục lục

[**1.** **Thông tin nhóm:** 3](#_Toc42203159)

[**1.1.** **Thông tin thành viên:** 3](#_Toc42203160)

[**1.2.** **Phân chia công việc:** 3](#_Toc42203161)

[**2.** **Đánh giá mức độ hoàn thành: 100%** 3](#_Toc42203162)

[**3.** **Quy trình xử lý trò chơi:** 4](#_Toc42203163)

[**4.** **Mô tả một số hàm quan trọng:** 4](#_Toc42203164)

[**4.1.** **Quy trình xử lý đọc đề random từ file:** 4](#_Toc42203165)

[**4.2.** **Các hàm xử lý đoán kết quả của người chơi:** 5](#_Toc42203166)

[**4.3.** **Các hàm xử lý lưu kết quả:** 6](#_Toc42203167)

[**4.3.1.** **Chuyển số về chuỗi** 6](#_Toc42203168)

[**4.3.2.** **Sắp xếp bảng xếp hạng và lưu** 6](#_Toc42203169)

[**5.** **Demo chương trình:** 7](#_Toc42203170)

[**5.1.** **Kiểm tra thông tin nhập tên người chơi:** 7](#_Toc42203171)

[**5.2.** **Máy random một đề từ file dethi.txt:** 8](#_Toc42203172)

[**5.3.** **Nếu người chơi giành chiến thắng:** 15](#_Toc42203173)

[**5.4.** **Nếu người chơi thua cuộc:** 18](#_Toc42203174)

[**5.5.** **Sau khi kết thúc trò chơi:** 19](#_Toc42203175)

[**6.** **Tài liệu tham khảo:** 20](#_Toc42203176)

# **Thông tin nhóm:**

## **Thông tin thành viên:**

|  |  |  |  |
| --- | --- | --- | --- |
| MSSV | Họ tên | Email | Số điện thoại |
| 18120429 | Phạm Trung Kiên | trungkien2000.tx@gmail.com | 0584175702 |
| 18120431 | A Ly Ha Kim | alyhakim6516@gmail.com | 0768006516 |
| 18120434 | Thái Hoàng Lâm | hoanglam372000@gmail.com | 0773956858 |
| 18120444 | Dương Thành Long | duongthanhlongk18hcmus@gmal.com | 0986189658 |
| 18120462 | Nguyễn Thị Mận | nguyenthiman2600@gmail.com | 0392164820 |

## **Phân chia công việc:**

|  |  |  |
| --- | --- | --- |
| Họ tên | Nội dung công việc | Tỉ lệ đóng góp (%) |
| Phạm Trung Kiên | Viết các hàm: random mã đề, lựa chọn cách chơi, nhập tên theo chuẩn, demo/test chương trình; viết báo cáo | 20% |
| A Ly Ha Kim | Viết các hàm: hàm khi chọn sai đáp án, các hàm xử lý khi đoán từ, ký tự, hỗ trợ hàm nhập tên, ghép chương trình, demo/test chương trình | 20% |
| Thái Hoàng Lâm | Viết các hàm: đọc đề từ file, làm các đề thi, các hàm Reset, IntToString (gọi nhiều trong chương trình), ghép chương trình | 20% |
| Dương Thành Long | Viết các hàm: hàm khi chọn đúng đoán đúng, các hàm xử lý khi đoán từ, ký tự, hỗ trợ ghi kết quả vào file, demo/test chương trình | 20% |
| Nguyễn Thị Mận | Viết các hàm: xuất thông báo, số lần đoán sai, các hàm xử lý đoán từ, ký tự, xuất đề trước và trong lúc chơi, so sánh, ghép chuỗi, ghi kết quả vào file, ghép chương trình | 20% |

# **Đánh giá mức độ hoàn thành: 100%**

* Nhập tên theo đúng chuẩn (các ký tự thuộc 0 – 9, A – Z, a – z, có thể chứa khoảng trắng trong tên): 100%
* Random đề từ file (chứa 100 đề, là 100 từ tiếng anh), xuất đề ở dạng ẩn (\*\*\*\*): 100%
* Lựa chọn phướng án trả lời (đoán từ hoặc đoán ký tự): 100%
* Xử lý “đoán từ” (xuất thông báo chiến thắng nếu đoán đúng, xuất thông báo thua cuộc và cho lựa chọn lại chơi tiếp hay ngừng): 100%
* Xử lý “đoán ký tự” (xuất ký tự trong đề nếu đoán đúng, xuất thông báo chiến thắng hay thua cuộc, lựa chọn chơi tiếp hay dừng): 100%
* Xuất trạng thái người chơi dưới dạng hình ảnh (sau mỗi lần đoán sai): 100%
* Lưu thông tin người chơi vào file (có sắp xếp thứ tự): 100%
* Lựa chọn chơi tiếp hay ngừng khi kết thúc trò chơi: 100%

# **Quy trình xử lý trò chơi:**

* Bước 1: người chơi nhập tên, theo đúng chuẩn được đặt ra (các ký tự trong tên nằm trong đoạn 0 – 9, A – Z, a – z, có thể chứa khoảng trắng trong tên)

Chương trình random đề trong file đề, dưới dạng các ký tự \*\*\*\*

* Bước 2: hiện 3 lựa chọn: Đoán từ | Đoán ký tự | Thoát
  + 2.1: Nếu chọn “Đoán từ”: người chơi nhập vào đáp án của cả từ
    - Nếu đúng: xuất đáp án ra màn hình, hỏi chơi tiếp hay thoát
    - Nếu sai: xuất đáp án, điểm ra màn hình, hỏi chơi mới, chơi lại hay thoát
      * Nếu chọn chơi lại, hệ thống sẽ ghi nhận tên cũ, và xóa điểm cũ, xuất bảng xếp hạng và lưu kết quả vào file.
      * Nếu chọn chơi mới, lưu dữ liệu vào file, xuất bảng xếp hạng, quay lại bước 1.
  + 2.2: Nếu chọn “Đoán ký tự”: mỗi lần sẽ nhập một ký tự người chơi đoán
    - Nếu đoán đúng ký tự ở lượt đó, xuất thông báo chọn “đoán từ” hay “đoán ký tự”. Nếu chọn “đoán từ”, đến bước 2.1, chọn “đoán ký tự” đến bước 2.2.
    - Nếu chọn sai, xuất hiện hình vẽ trạng thái người chơi, xuất thông báo chọn “đoán từ” hay “đoán ký tự”. Nếu chọn “đoán từ”, đến bước 2.1, chọn “đoán ký tự” đến bước 2.2.
    - Nếu sau 7 lần đoán sai, xuất hiện thông báo gồm tên, điểm, số lần đúng. Quay lại bước 2.
    - Nếu đoán đúng trước 7 lần, xuất đáp án. Quay lại bước 2
  + 2.3: Nếu chọn “Thoát”: xuất tên, điểm, số lần đúng xuất bảng xếp hạng, lưu vào file, kết thúc trò chơi

# **Mô tả một số hàm quan trọng:**

## **Quy trình xử lý đọc đề random từ file:**

* Bước 1: chạy hàm \_Random số từ 1 – 100 (do có 100 đề)  
  Các xử lý trong hàm \_Random: số random được lưu ở v1
  + Lấy thời gian của hệ thống (bằng lệnh của Java)
  + Dùng hàm lấy random số trong khoảng 0 – n của Java, lưu ở a0
  + Tăng mỗi giá trị random thêm 1 (do hàm chạy kết quả random ra từ 0 -99), vì thứ tự đầu tiên của đề bắt đầu từ 1 (nếu không tăng thêm 1, có thể random ra 0)
  + Lưu v1 bằng giá trị của a0
* Bước 2: nhảy đến hàm \_ReadFile để đọc toàn bộ nội dung file, lưu toàn bộ nội dung trong file thành một chuỗi, lưu ở biến str
* Bước 3: sau khi đọc được file, nhảy đến hàm \_LayDe, để lấy đề tương ứng với số random được ở bước 1  
  Các xử lý trong hàm \_LayDe:
  + Lưu str vào a0, str1 là đề sẽ lấy, lưu vào s4
  + Nhảy đến hàm \_getString: mục đích để lấy toàn bộ phần chuỗi bắt đầu từ vị trí được random ở bước 1 (lưu giá trị random, lúc này là v1 vào s6)  
    Các xử lý trong hàm \_getString: duyệt qua từng ký tự trong chuỗi str, nếu ký tự là ‘\*’, tăng giá trị biến đếm t0. Dừng vòng lặp khi giá trị biến đếm t0 = s6. Chuỗi còn lại từ vị trí đang đếm, lưu vào s0.

Lúc này, s0 đang lưu phần chuỗi được lấy từ file, với từ đầu tiên là từ có thứ tự được random ở bước 1.

* + Nhảy đến hàm \_Get\_test: hàm này dùng để lấy đề, chính là từ đầu tiên của chuỗi đang lưu ở s0  
    Các xử lý của hàm \_Get\_test: lấy từng ký tự của s0, lưu vào s4. Tăng địa chỉ s0, tăng biến đếm, tiếp tục lưu từng ký tự đến ghi gặp ký tự ‘\*’ hoặc ‘\0’ thì dừng.  
    Đề lúc này được lưu ở s4.

## **Các hàm xử lý đoán kết quả của người chơi:**

* Đoán từ:
  + Bước 1: khi nhận chuỗi được nhập từ người dùng, nhảy đến hàm SoSanhChuoi
  + Bước 2: hàm SoSanhChuoi thực hiện thao tác so sánh từng ký tự của 2 chuỗi.  
    Quy trình xử lý của hàm (trả về v1 = 1 nếu t8 > t9, v1 = -1 khi t8 < t9, v1 = 0 khi t8 = t9):  
    - Tham số đầu vào của 2 chuỗi là t8, t9  
    - Gọi hàm \_DemKyTu cho 2 chuỗi, lưu vào a1, a2 (nếu a1 < a2 -> trả về v1 = -1, nếu a1 > a2 -> trả về v1 = 1)  
    - Nếu a1 = a2, thực hiện so sánh từng cặp ký tự với t0, t1, lần lượt là ký tự của 2 chuỗi, nếu tồn tại t0 > t1 -> v1 = 1, nếu t0 < t1 -> v1 = 0.   
    - Chạy đến hết chuỗi -> v1 = 0
  + Bước 3: kết thúc quá trình so sánh, nếu v1 = 0 -> người chơi thắng. Ngược lại, người chơi thua.
* Đoán ký tự:
  + Bước 1: khi nhận được ký tự người chơi nhập vào, lưu ký tự đó vào t0
  + Bước 2: chuyển tiếp đến hàm \_Doan1KyTu\_Lap: xử lý việc lặp qua từng ký tự trong chuỗi đề
  + Bước 3: có các trường hợp
    - Nếu gặp ký tự ‘\n’: kết thúc vòng lặp
    - Nếu gặp ký tự đã có ở str2: nhảy đến HamSai
    - Nếu đúng: tăng địa chỉ, gán ký tự vào str2, nhảy đến HamDung

HamSai: nhảy đến vị trí kết thúc đoán ký tự, tăng số lần đoán sai  
HamDung: gán ký tự vào str2, tăng địa chỉ các biến, nhảy lại bước 2­­

## **Các hàm xử lý lưu kết quả:**

### **Chuyển số về chuỗi**

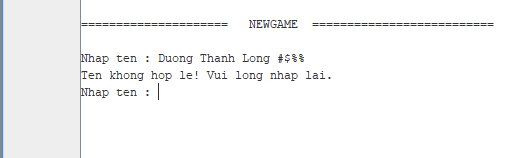
* Hàm IntToString:
  + Nhận đầu vào là v1 (số cần chuyển đổi)
  + Chia 10 lấy dư, load vào str4 (dừng khi thương = 0)
* str4 chứa chuỗi đảo ngược của số. VD: số 126 -> str4 = ‘621’
* \_Reversion\_string: (hàm đảo ngược chuỗi)
  + Địa chỉ chuỗi chứa kết quả của của hàm được lưu trong t7
  + Hàm load lần lượt các ký tự số của str4 vào chuỗi kết quả
* Chuỗi ký tự đúng. VD số 126 -> str4 = ‘621’ -> ‘126’

### **Sắp xếp bảng xếp hạng và lưu**

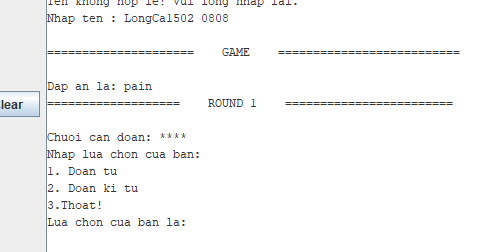
* Hàm \_BXH1: đọc chuỗi trong file ‘nguoichoi.txt’ lưu vào biến BXH1
* Hàm GhepChuoi: ghép chuỗi name (tên người chơi), str5 (số điểm) và str4 (số lần đúng) về dạng Tên\_Điểm\_Sốlầnđúng
* Hàm ViTriSX: trả về vị trí người chơi trong BXH
  + Đưa tham số vào:   
    la $s0, BXH1   
    la $s1, str5
  + Các biến đếm:   
    li $t0, 0  
    t0 = 10  
    li $t1, 1  
    t0 đếm số ký tự ‘\n’, và dừng khi t0 = 10, t1 là vị trí trả về
  + Tim\_: dùng tìm ký tự ‘\_’ trong BXH1, khi gặp ký tự ‘\_’, nhảy tới nhãn TachDiem
  + TachDiem: lưu chuỗi điểm vào str4, khi gặp ký tự ‘\_’, nhảy tới nhãn SoSanh
  + SoSanh: str4 là điểm số của người chơi cũ; str5 là điểm số hiện tại, gọi hàm SoSanhChuoi (str4, str5). Nếu v1 != 1(str4 <= str5) thì kết thúc trả về vị trí. Nếu v1 = 1 -> tăng t0, t1 thêm 1 (+1), quay lại nhãn Tim\_
  + Lưu kết quả vào v1
* TaoBangMoi: lưu BXH mới, vào BXH2
  + t3 = 0, khởi tạo biến đếm, dừng khi t3 = 11
  + Lưu trước: tăng biến đếm, load các vị trí trước v1 vào BXH2
  + LuuKQ: tăng biến đếm, load kết quả vào BXH2
  + Lưu sau: tăng biến đếm, load các vị trí sau v1 vào BXH2
* Luu\_SX: gọi các hàm GhepChuoi, \_BXH1, ViTriSX, TaoBangMoi
* Xuất BXH
* GhiF: ghi BXH2 vào fout

# **Demo chương trình:**

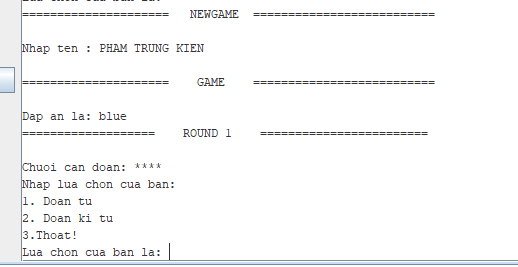
## **Kiểm tra thông tin nhập tên người chơi:**



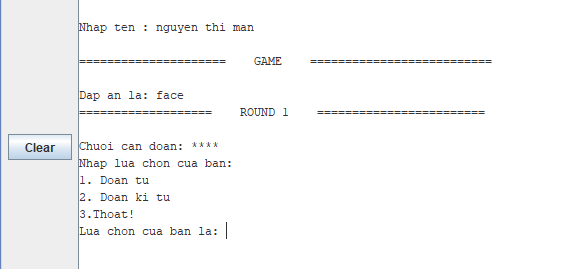
*tên chứa các ký tự đặc biệt -> sai, yêu cầu nhập lại*



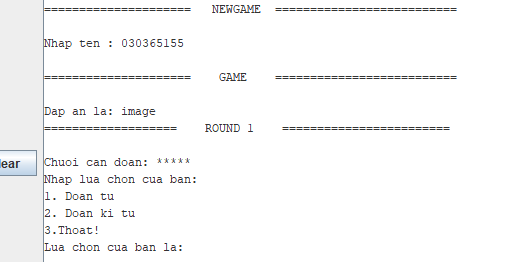
*tên chứa số và các chữ cái hoa, thường, dấu cách -> hợp lệ*



*tên chỉ chứa kí tự hoa, dấu cách -> hợp lệ*



*tên chỉ chứa kí tự thường, dấu cách -> hợp lệ*

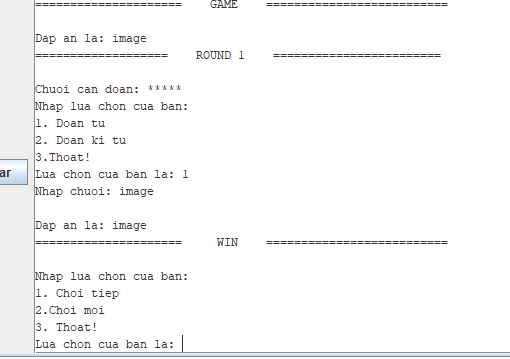


*tên chỉ chứa số -> hợp lệ*

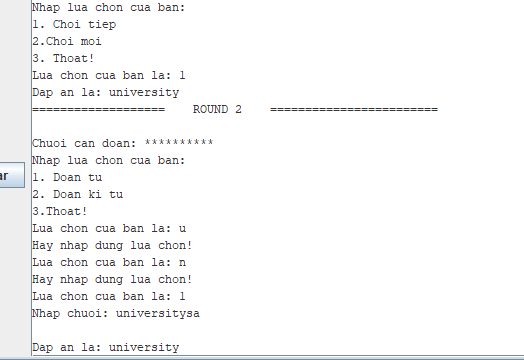
## **Máy random một đề từ file dethi.txt:**

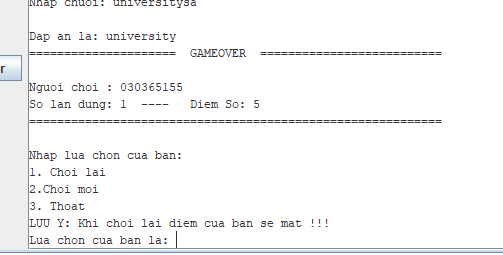
*xuất dưới dạng ẩn (\*\*\*\*), đồng thời có các lựa chọn cho người chơi chọn:*

* Lựa chọn đoán từ:



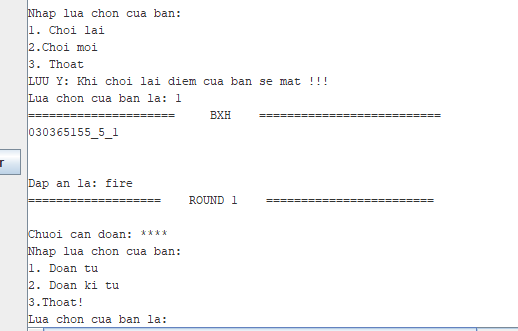
*lựa chọn đoán từ đúng*



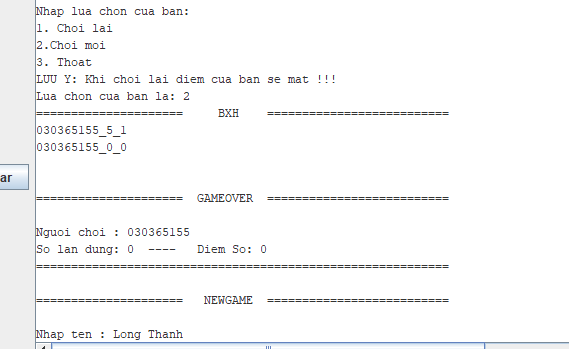


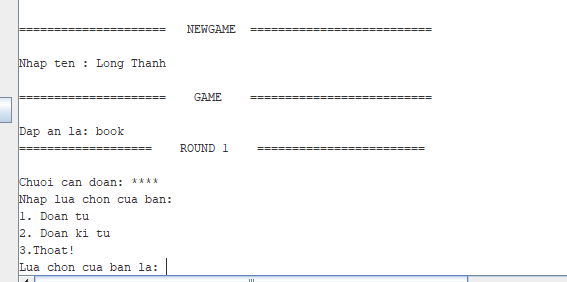
*lựa chọn đoán từ sai*

*xuất hiện các lựa chọn sau khi GAMEOVER: nếu chọn chơi lại sẽ lưu kết quả và xuất bảng xếp hạng, nếu chọn chơi mới thì kết thúc lượt chơi của người chơi hiện thời và xuất ra bảng xếp hạng hiện thời*

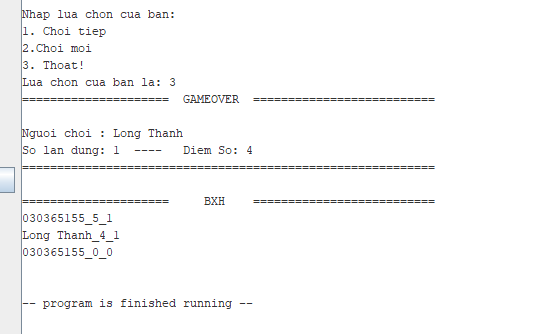


*trường hợp lựa chọn chơi lại*



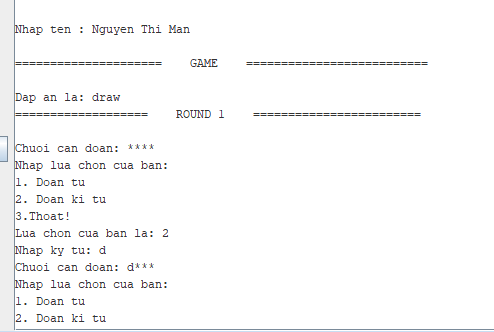


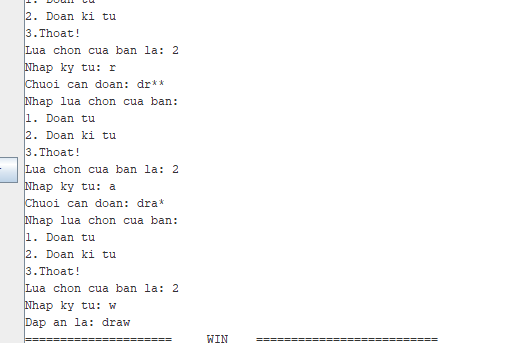
*trường hợp khi lựa chọn chơi mới*



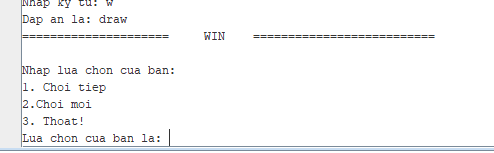
*trường hợp chọn thoát: kết quả được lưu và xuất bảng xếp hạng*

* Lựa chọn đoán ký tự
  + Đoán đúng



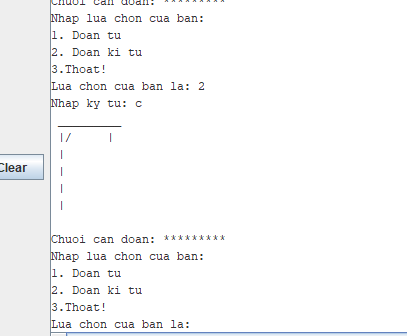


*đoán đúng ký tự*

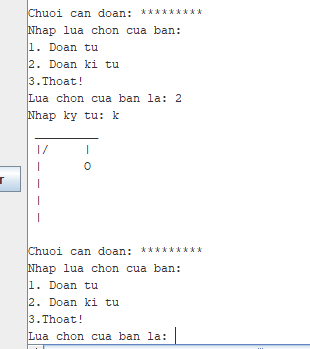


*đoán đúng hết ký tự trong đề (nếu sau đó chọn ngoài 3 lựa chọn được đưa ra -> nhập lại)*

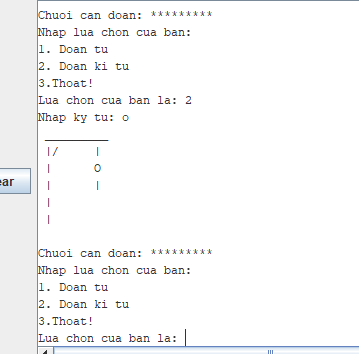
* + Đoán sai



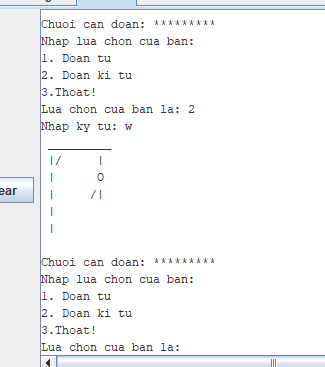
*sai lần 1*



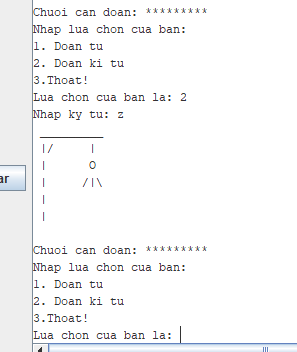
*sai lần 2*



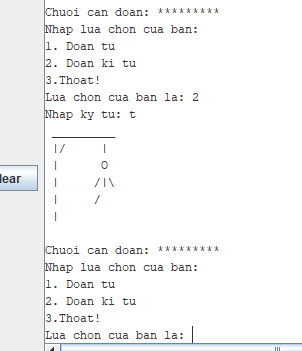
*sai lần 3*



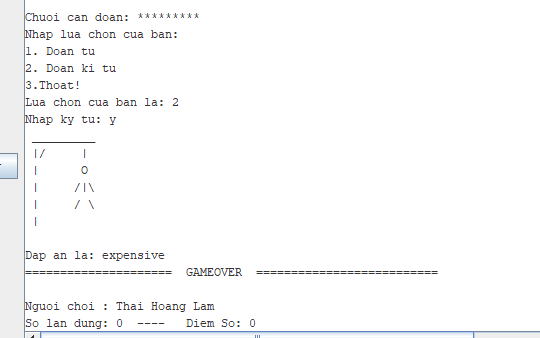
*sai lần 4*



*sai lần 5*



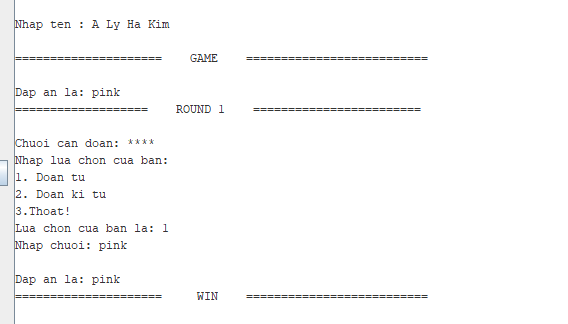
*sai lần 6*

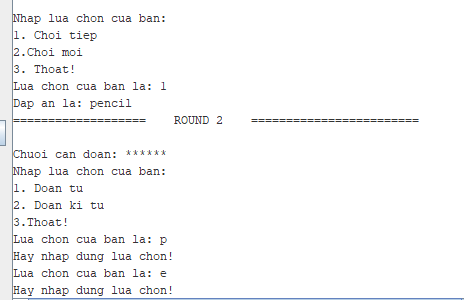


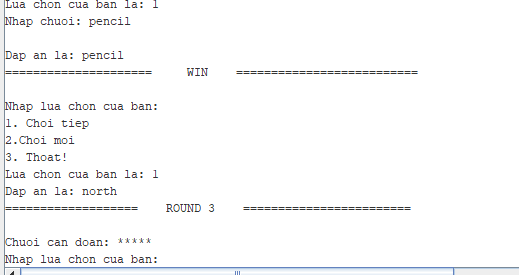
*sai lần 7*

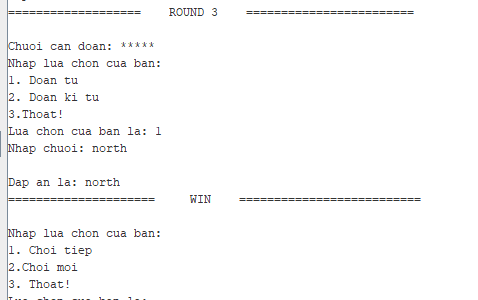
## **Nếu người chơi giành chiến thắng:**

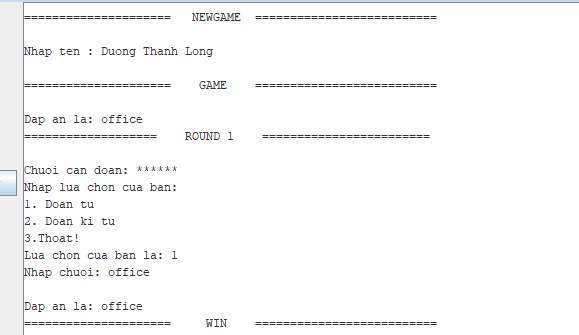
*người chơi sẽ được cộng số điểm bằng đúng với số ký tự của Đề Thi đã cho, tăng số lần đúng, round chính là số lần đúng. Máy sẽ random 1 đề thi mới khác với các đề thi đã cho và bắt đầu lượt chơi mới (các đề thi có thể trùng nhau)*

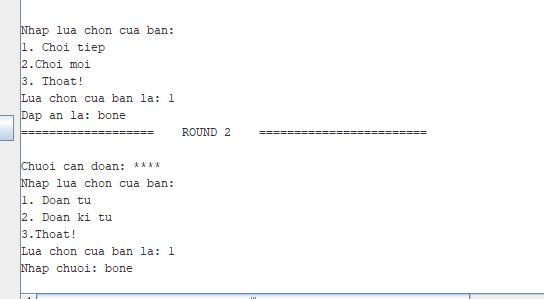


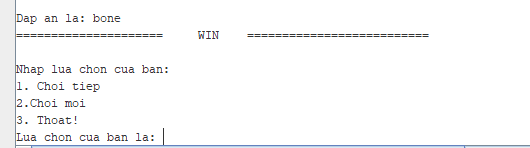


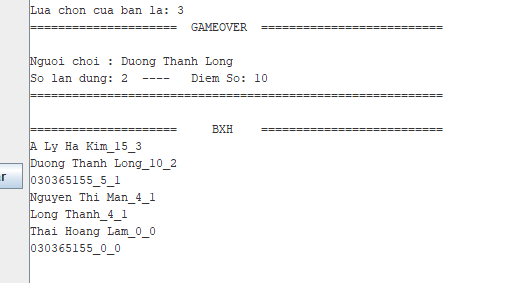






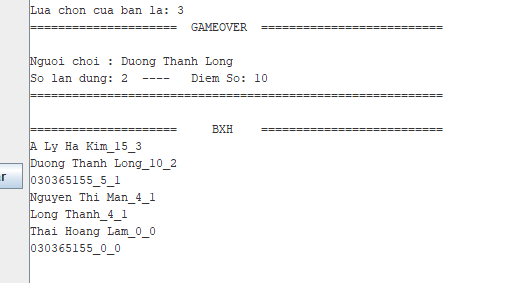


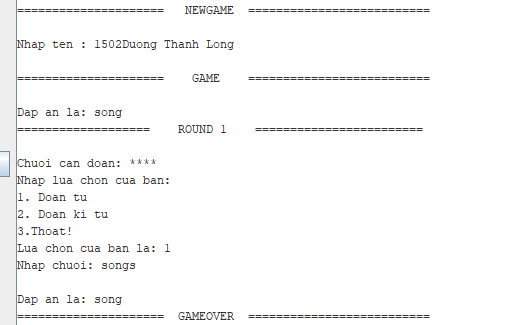


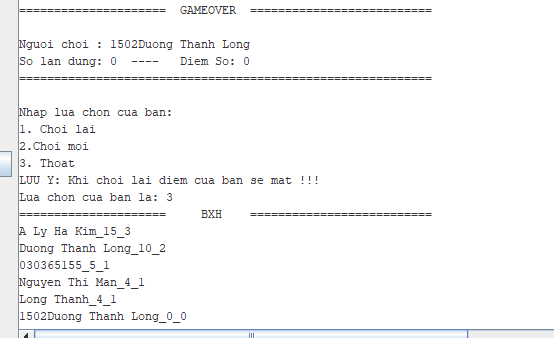


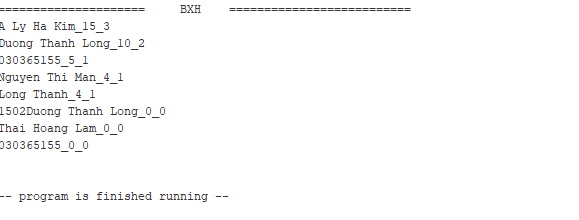
## **Nếu người chơi thua cuộc:**

*máy sẽ xuất thông báo gồm các thông tin: Tên người chơi – Tổng số điểm – số lượt chiến thắng và lưu các thông tin này vào tập tin nguoichoi.txt*



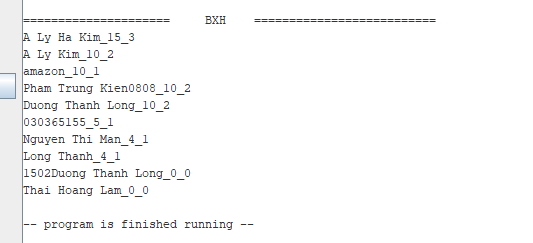




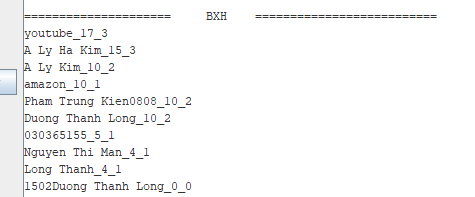


## **Sau khi kết thúc trò chơi:**

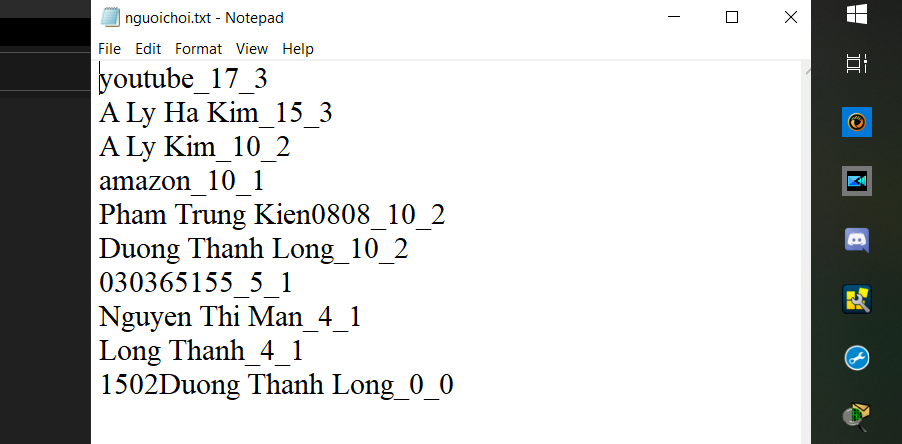
*máy sẽ xuất danh sách 10 người chơi có số điểm cao nhất trong tập tin nguoichoi.txt đã lưu trữ*



*bảng xếp hạng trước khi có người chơi 11 (youtube)*



*bảng xếp hạng sau khi có người chơi 11 (youtube)*



*File nguoichoi.txt*

# **Tài liệu tham khảo:**

* <https://courses.missouristate.edu/KenVollmar/MARS/Help/SyscallHelp.html>
* Bài giảng và code demo của giảng viên hướng dẫn  
  *chú thích: với các vị trí ghi s0, s1.. t0, t1.. tương ứng là thanh ghi $s0, $s1,.. $t0, $t1…*