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

**TRƯỜNG ĐẠI HỌC KHOA HỌC TỰ NHIÊN**

**KHOA CÔNG NGHỆ THÔNG TIN**



**BÁO CÁO THỰC HIỆN ĐỒ ÁN PROCESS 2**

NHÓM 2 – LỚP 19CLC8

**GV HƯỚNG DẪN: THÁI HÙNG VĂN**

*Thành phố Hồ Chí Minh - 2021*

**THÔNG TIN VÀ PHÂN CÔNG THÀNH VIÊN NHÓM**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **MSSV** | **Họ tên** | **Email** | **Phân công** | **Tiến độ hoàn thành** |
| 19127083 | Nguyễn Hữu Tuấn | 19127083@student.hcmus.edu.vn | UI của máy phụ huynh | 100% |
| 19127100 | Lê Trần Gia Bảo | 19127100@student.hcmus.edu.vn | Xử lí xung đột + xử lí đồng bộ, xử lí đa luồng | 100% |
| 19127294 | Nguyễn Trần Thiện Toàn | 19127294@student.hcmus.edu.vn | Xử lý các tính năng trên máy trẻ em (đọc, cập nhật thông tin, kiểm tra tắt máy, log off màn hình, kiểm tra thời gian sử dụng, tạo các Class | 100% |
| 19127363 | Lê Văn Đông | 19127363@student.hcmus.edu.vn | Thực hiện các chức năng của windows như shutdown, tắt màn hình, chụp màn hình, | 100% |

Đánh giá tiến độ hoàn thành: 100%

**MỤC LỤC**

[1. Về xử lí cho máy trẻ em, người bị theo dõi (Chương trình C) 1](#_Toc77194915)

[a. Các Lớp đối tượng và cấu trúc được sử dụng chính: 1](#_Toc77194916)

[b. Các hàm xử lý máy trẻ em 4](#_Toc77194917)

[c. Xử lý đa luồng 5](#_Toc77194918)

[d. Cơ chế hàm main() 6](#_Toc77194919)

[2. Về xử lí cho máy người lớn, người theo dõi (Chương trình P) 7](#_Toc77194920)

[a. Về giao diện người dùng 7](#_Toc77194921)

[i. Class TextObject 7](#_Toc77194922)

[ii. Nhóm hàm chức năng 9](#_Toc77194923)

[iii. Nhóm hàm thực hiện chức năng Nhập/Xuất 10](#_Toc77194924)

[iv. Nhóm hàm show Menu 12](#_Toc77194925)

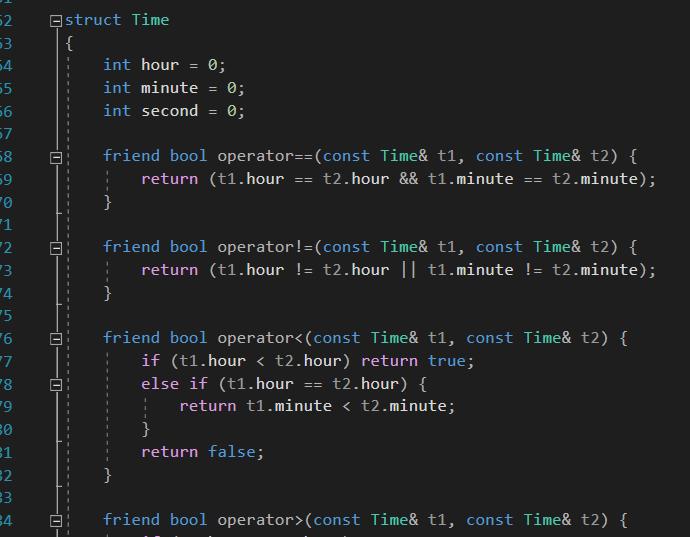
[b. Về xử lí critical section 21](#_Toc77194926)

[**THAM KHẢO** 23](#_Toc77194927)

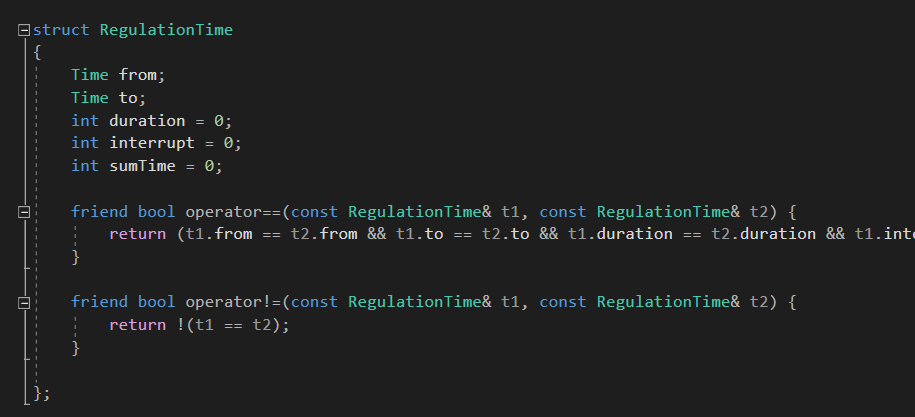
# Về xử lí cho máy trẻ em, người bị theo dõi (Chương trình C)

## Các Lớp đối tượng và cấu trúc được sử dụng chính:

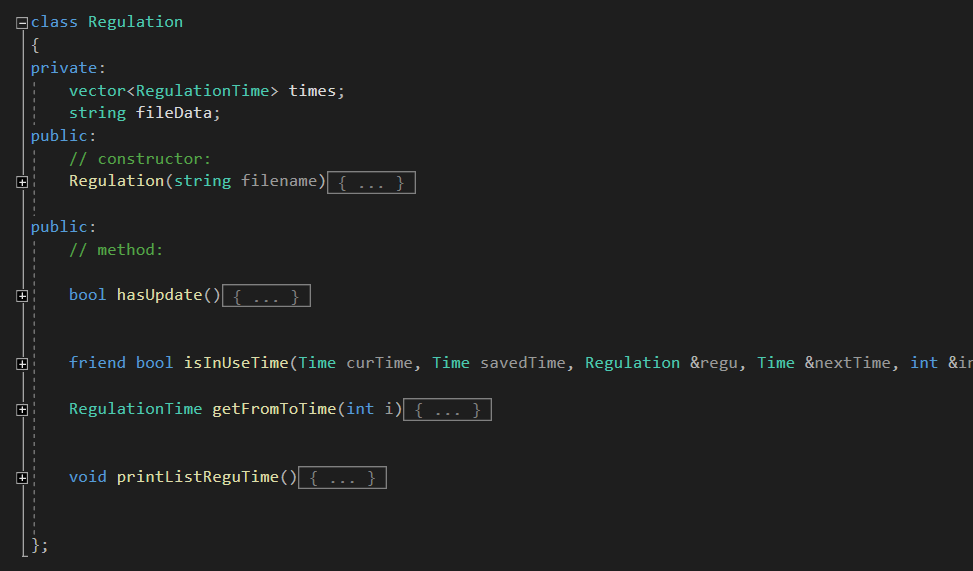
* Struct Time: kiểu dữ liệu lưu trữ thời gian (giờ phút, giây). Cấu trúc này được cài thêm một số operator để so sánh và tính toán.



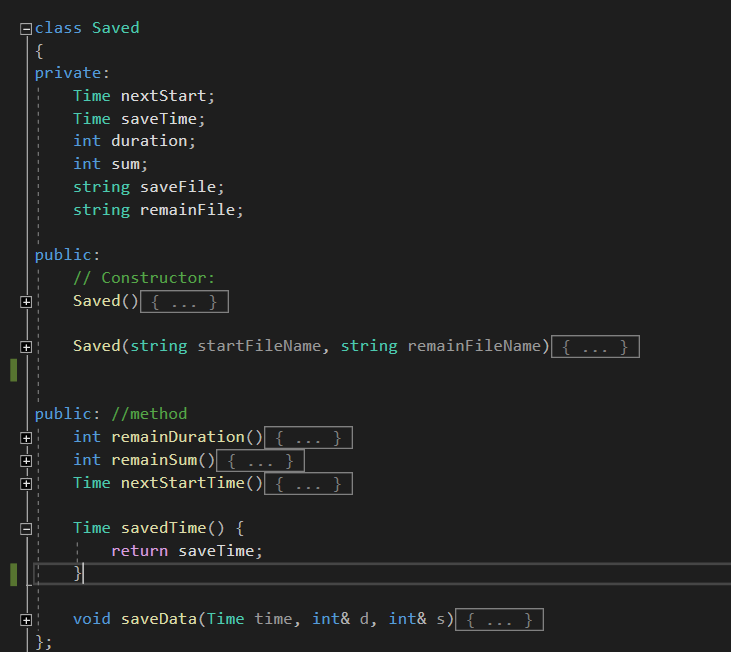
* Struct RegulationTime: Cấu trúc dùng để lưu những thông tin quy định về thời gian sử dụng.



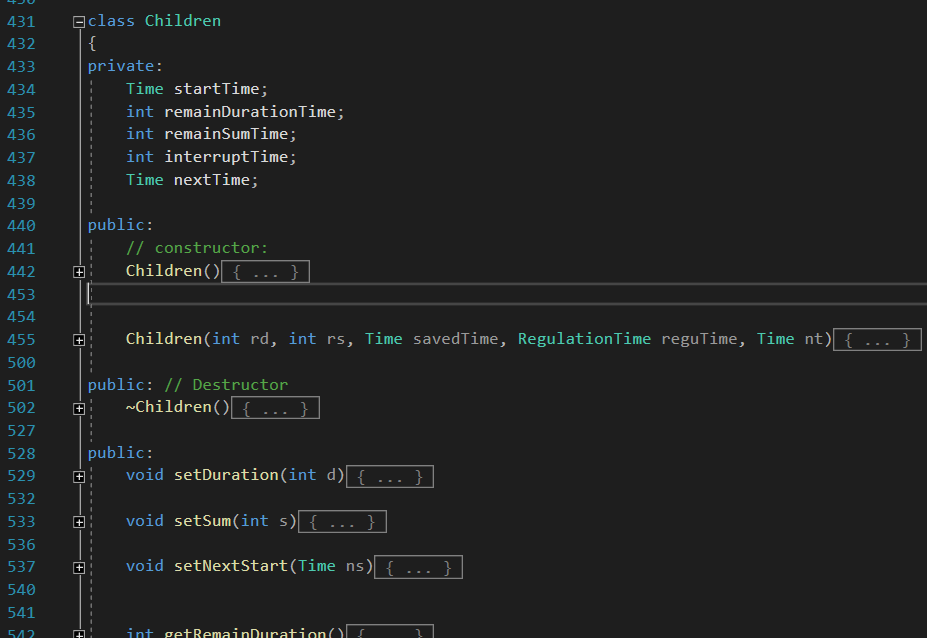
* Class Regulation: class này là một tập hợp các thông tin RegulationTime. Các phương thức chủ yếu là đọc data quy định, kiểm tra data có cập nhật không.



* Class Saved: lớp được dùng để lưu các thông tin đã dùng của trẻ như thời gian còn lại được phép sử dụng. Các thông tin về thời gian lưu, thời gian được sử dụng tiếp theo nếu trẻ bị phạt. Thông tin được lấy từ 2 file. Đó là savedStart - thời gian trẻ được phép mở máy (nếu có bị phạt, nếu không có mặc định là thời gian bắt đầu khung giờ hiện tại); file còn lại saveRemain, thông tin chính ở đây là tổng thời gian được sử dụng còn lại trong khung giờ.



* Class Children: Đây là class rất quan trọng. Nó giữ tất cả các thông tin về thời gian, khung giờ. Lớp sẽ được tạo dựa trền thông tin khung giờ đang được sử dụng cùng vời những thông số cũ của khung giờ đó (trong class Saved - nếu có). Các phương thức chủ yếu là update() để cập nhật trạng thái hiện tại, countDown để trừ đi thời gian sử dụng. Destructor được tạo nhằm lưu thông tin của quá trình vừa dụng khi chương trình kết thúc.



## Các hàm xử lý máy trẻ em

* Hàm getDirCapture() dùng để lấy đường dẫn mặc định, không có tham số. Kết quả trả về kiểu string là đường dẫn: “./history/năm\_tháng\_ngày/giờ\_phút.png”, thời gian năm, tháng, ngày, giờ, phút là thời gian hệ tại của hệ thống.
* Hàm captureScreen(string dir). Hàm này dùng để chụp màn hình. Tham số truyền vào là đường dẫn và tên file. Ta có thể dùng hàm getDirCapture() dùng làm tham số.
* Hàm turnoffScreen(int second) hàm này nhằm để tắt màn hình trong second giây. Khi gọi hàm, màn hình sẽ tắt tạm thời, nhằm không cho trẻ em thao tác trên màn hình. Khi hết thời gian tắt màn hình, trẻ có thể tiếp tục sử dụng chương trình.
* Hàm scaleIMG(string des, int height, int width). Chức năng của hàm này nhằm thay đổi kích thước của hình ảnh lại để hiển thị trên GUI của phụ huynh. des là đường dẫn + tên file ảnh cần lưu, height là chiều cao của hình, width là chiều rộng của hình.
* Hàm getTime() sẽ trả về kiểu dữ liệu Time bao gồm giây, phút, giờ hiện tại.
* Hàm MySystemShutdown(): hàm này tận dụng Window API của hệ điều hành để tắt máy.
* isInUseTime(): hàm kiểm tra xem liệu có đang trong thời được phép xử dụng hay không. Thời gian để so sánh được lấy từ cloud và những thông tin thời gian đã sử dụng trước đó (sum, có bị phạt 10 phút không,…). Đồng thời hàm sẽ giúp biết được khung giờ kế tiếp là mấy giờ.
* InputPassword(): hàm sẽ gọi hàm kiểm tra isInUseTime() và kiểm tra đây là ba mẹ hay trẻ em.
* CheckUpdate(): hàm sẽ gọi hàm update của class Regulation. Nếu có thay đổi sẽ sẽ gọi tiếp các hàm update của các class khác đồng thời kiểm tra luôn liệu còn được sử dụng không.

## Xử lý đa luồng

* Hai thread được sử dụng để thực hiện song song việc đếm 15s và nhập mật khẩu là:
* Thread t1(inputThread,BAME): thread với đối số truyền vào là hàm inputThread(). Hàm này sẽ cho người dùng nhập pass và chỉ dừng khi người dùng nhập đúng pass phụ huynh. Sau đó nó sẽ đặt biến toàn cục isLogin bằng true.
* Threadt2(countDown,ref(countDownTime)): thread với đối số truyền vào là hàm contDown(int& time). Hàm này sẽ liên tục đếm ngược cho đến khi biến time bằng 0, sau đó nó sẽ kiểm tra biến isLogin nếu bằng false se thông báo người dùng bật lại máy sau 10 phút và shutdown hệ điều hành.
* Lưu ý, ở hàm main() sau khi tạo hai thread trên ta join thread t1 trước để chương trình đợi thread t1 kết thúc. Nếu t1 nhập đúng pass phụ huynh trước 15s thì ta sẽ join thread t2 để ngắt việc shutdown. Nếu t1 nhập sai pass hoặc hết thời gian timeout thì t2 sẽ shutdown hệ điều hành.
* Ba thread được sử dụng để thực hiện song song việc chụp màn hình, kiểm tra thay đổi của file môi phút và thông báo khi thời gian sử dụng còn 1 phút là:
* Thread capture(captureScreen): thread với đối số truyền vào là hàm captureScreen(). Hàm này sẽ chụp lại màn hình và lưu vào folder, sau đó lặp lại mỗi phút và chỉ dừng khi tổng thời gian sử dụng còn lại bằng 0.
* Thread check(checkUpdateData): thread với đối số truyền vào là hàm checkUpdateData(). Hàm này sẽ kiểm tra mỗi phút trạng thái update bằng hàm checkUpdate(), hàm này sẽ kiểm tra và return giá trí vào biến updateStatus. Nếu status = 0 – không có update, status = 1 - có update và thông báo cho người dùng thời gian còn lại, status = -1 - có update và đã hết thời gian sử dụng sau đó shutdown hệ điều hành. Hàm checkUpdateData() chỉ dừng khi tổng thời gian sử dụng còn lại bằng 0.
* Thread notify(notificate()): thread với đối số truyền vào là hàm notificate(). Hàm này sẽ lặp lại mỗi phút và trừ thời gian còn lại và tổng thời gian còn lại đi 1 phút. Hàm sẽ kiểm nếu thời gian còn lại đã hết sẽ tắt màn hình và nghỉ trong thời gian interupt cho trước, nếu thời tổng thời gian còn 1 phút thì sẽ thông báo cho người dùng.
* Ở main, ta sẽ join cả 3 thread trên, main se đợi cả 3 thread thực thi xong. Khi đó, main sẽ shutdown hệ điều hành vì cả 3 thread chỉ dừng khi tổng thời gian còn lại bằng 0.

## Cơ chế hàm main()

* Đầu tiên một đối tượng Regulation được khởi tạo dựa trên thông tin quy định. Sau đó đến Saved, Children(tre). Các biến toàn cục như gNextTime cũng được sử để tiện cho việc theo dõi. Một vòng lặp do while được thực hiện, nó chỉ lặp lại nếu ứng với người dùng là ba mẹ. Đầu tiên nó sẽ thông báo nhập password. Biến status lưu trạng thái trả về của hàm nhập pass.
* Ứng với status = 1: cha mẹ đang dùng máy của trẻ. Việc kiểm tra chỉ thực hiện sau N\*phút (trong code là 2 phút).
* Case = 0: không trong khuôn giờ cho phép. Lúc này một hộp thoại sẽ thông báo đến mấy giờ mới được dùng lại. Đồng thời nó sẽ tạo ra 2 tiến trình song song. Một tiến trình kiểm tra coundown trong 15s để tắt máy, một cái để kiểm tra nếu nhập đúng pass ba me thì tiếp tục bình thường.
* Case = -1: Nhập đúng pass trẻ em và trong khung giờ sử dụng. Thông báo còn bao nhiêu phút và tạo ra 3 tiến trình. Các tiến trình kiểm tra chụp hình sau mỗi phút, kiểm tra cập nhật, kiểm tra hết giờ.
  + Case = -2: nhập sai 3 lần và tắt máy.

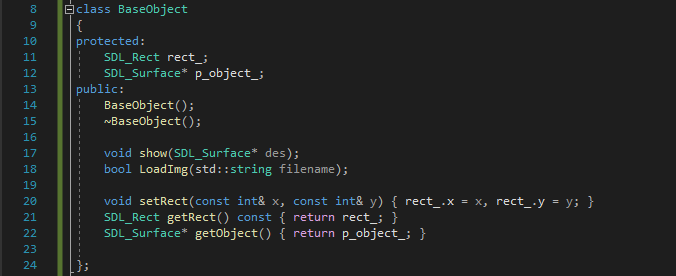
# Về xử lí cho máy người lớn, người theo dõi (Chương trình P)

## Về giao diện người dùng

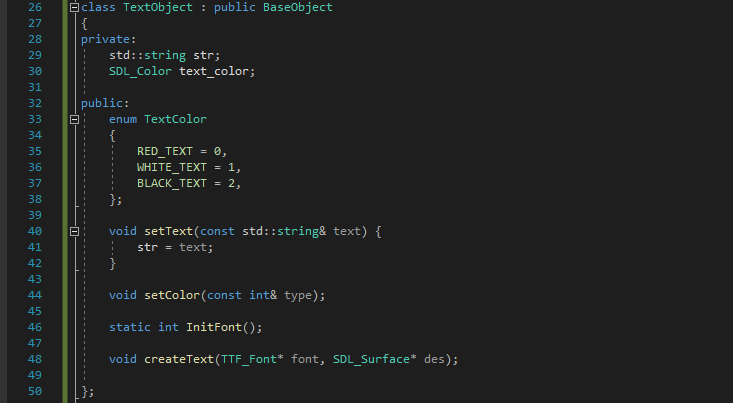
* Giao diện người dùng được cài đặt bằng cách sử dụng thư viện SDL để thiết kế đồ họa
* Về phần code, giao diện người dùng được chia ra làm 3 phần lớn: Các hàm phụ trợ, nhóm hàm Menu và nhóm hàm xử lí Input/Output.
* Ngoài ra, ta cần một lớp dành cho việc hiển thị Text lên màn hình, gọi là Class::TextObject

### Class TextObject

* Class TextObject được viết theo hướng OOP, được kế thừa từ Class BaseObject với 2 biến cơ bản là rect\_ - lưu trữ tọa độ của Object và p\_object\_ lưu trữ dữ liệu có dạng SDL\_Surface – dùng để hiển thị nội dung lên màn hình với các chức năng cơ bản như get, set. Ngoài ra, còn có hàm LoadImg để tải dữ liệu (hình ảnh, text, …) lên biến để hiển thị, ở đây là biến p\_object\_. Hàm show để hiển thị hình ảnh lên màn hình console.



* Class TextObject với hàm nổi bật là hàm createText, với chức năng để hiển thị chữ lên màn hình

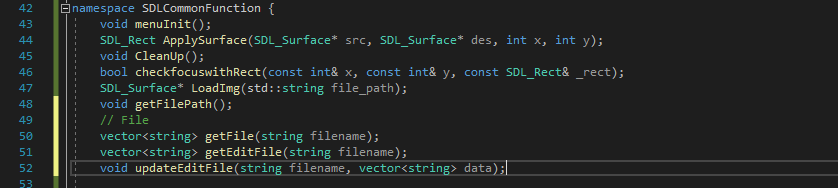


* Dưới đây là hàm createText():

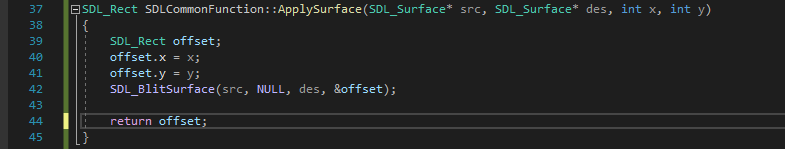


### Nhóm hàm chức năng

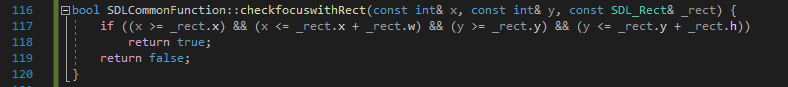
* Nhóm hàm chức năng được đưa vào một namespace tên SDLCommonFunction, để tránh trùng lặp với các hàm được thêm vào sau này.
* Nhóm hàm chức năng gồm các hàm sau đây:



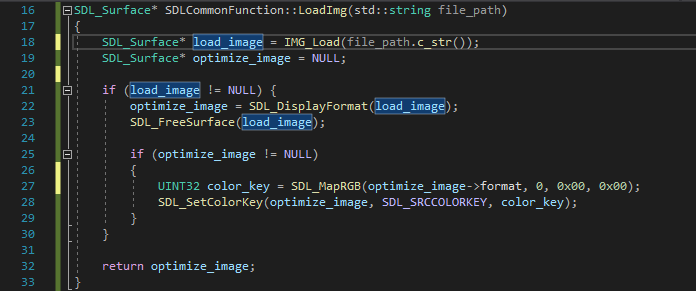
* Hàm nổi bật của nhóm hàm này là ApplySurface(); checkfocuswithRect() và LoadImg().
  + Với hàm ApplySurface(), hàm này đưa object - ở đây là src lên console - ở đây là biến des ở tọa độ (x,y) và trả về tọa độ của object mới đưa lên (bao gồm width – w và height – h)



* + Với hàm checkfocuswithRect(), hàm này để kiểm tra xem trỏ chuột hiện tại có nằm trong vùng obbject hay không (dùng để click vào một title nào đó)



* + Với hàm LoadImg(), hàm này được áp dụng để tải hình (từ file .png, .jpg, .bmp, …) và lưu vào biến SDL\_Surface, sau đó có thể đưa lên màn hình nhờ hàm ApplySurface()

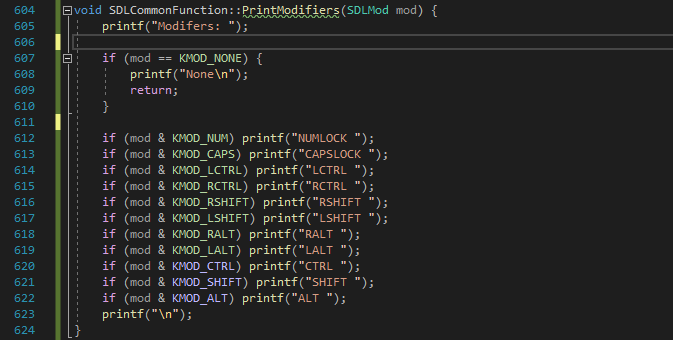


### Nhóm hàm thực hiện chức năng Nhập/Xuất

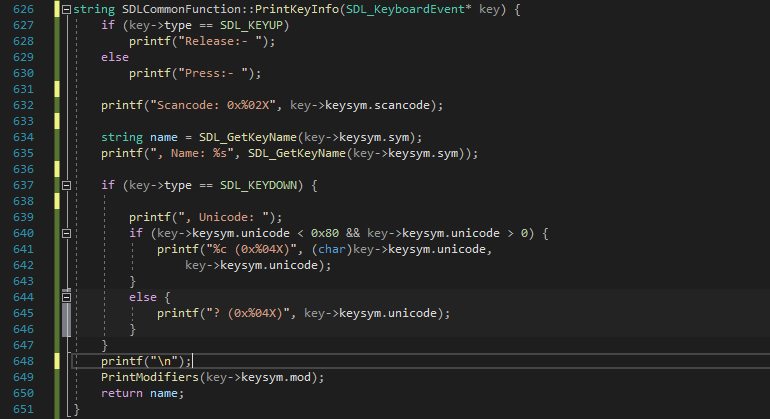
* Nhóm hàm này bao gồm 2 hàm, đó là PrintModifiers() và PrintKeyInfo(), có chức năng nhận biết mỗi khi người dùng thực hiện việc Input từ bàn phím (bởi vì SDL không có chức năng nhập từ bàn phím nên ta cần hàm này để thực hiện việc nhập, xuất từ bàn phím)



* + Hàm PrintModifiers(), có chức năng in lên console những phím chức năng (có thể kể đến như Shift, Enter, Caps lock, …), từ đó ta có thể xử lí việc nhập xuất

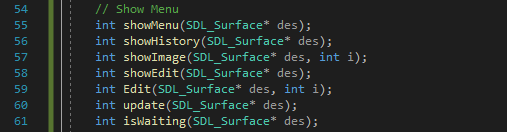


* + Hàm PrintKeyInfo(), cho ta biết thông tin tất cả các phím nhập từ bàn phím, có thể tham khảo link sau để biết tên của các phím: [SDLKey (libsdl.org)](https://www.libsdl.org/release/SDL-1.2.15/docs/html/sdlkey.html#SDLMOD)

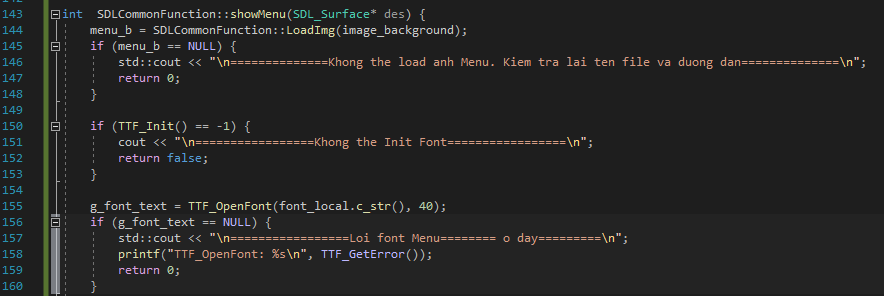


### Nhóm hàm show Menu

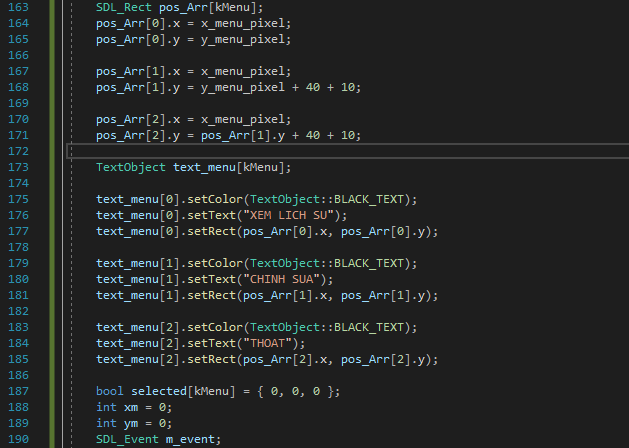
* Nhóm hàm show Menu gồm rất nhiều hàm, tuy nhiên nổi bật nhất là hàm showMenu() và Edit(), còn lại thì tương tự như 2 hàm trên, tùy vào chức năng mà ta sẽ code sao cho phù hợp. Dưới đây là các hàm:



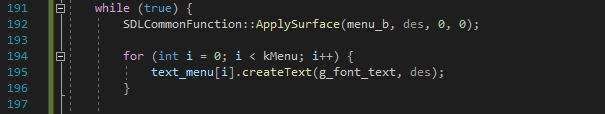
* Hàm showMenu() có công dụng hiển thị menu; showHistory() có nhiệm vụ hiển thị một list hình ảnh để người dùng có thể click vào để xem; showImage() để hiển thị hình ảnh mà người dùng click vào; showEdit() để hiển thị màn hình gồm các phần mà người dùng muốn chỉnh sửa (hoặc thêm mới); Edit() để chỉnh sửa nội dung (ở đây là dòng I trong file) người dùng vừa bấm vào ở hàm showEdit(); hàm update() để cập nhập nội dung người dùng thêm mới vào file và hàm isWaiting() để hiển thị cho người dùng biết có người đang chỉnh sửa chung với mình hay không.
  + Hàm showMenu():
    - Trước tiên, ta cần khởi tạo biến phông nền và font chữ cho chữ được hiển thị trên Menu



* + - Sau đó, ta khởi tạo các biến cần thiết, mà ở đây là tọa độ, nội dung. Biến selected để kiểm tra con trỏ chuột có nằm trong vùng chữ hoặc việc bấm chuột có bấm vào trong vùng có chữ hay không. Biến m\_event cho ta lấy được các sự kiện diễn ra trong khi chương trình đang chạy (có thể kể đến việc nhập/xuất, việc di chuyển chuột, việc bấm chuột, …)



* + - Tiếp theo, trong vòng while, ta sẽ cho hiển thị những thứ vừa setup lên màn hình



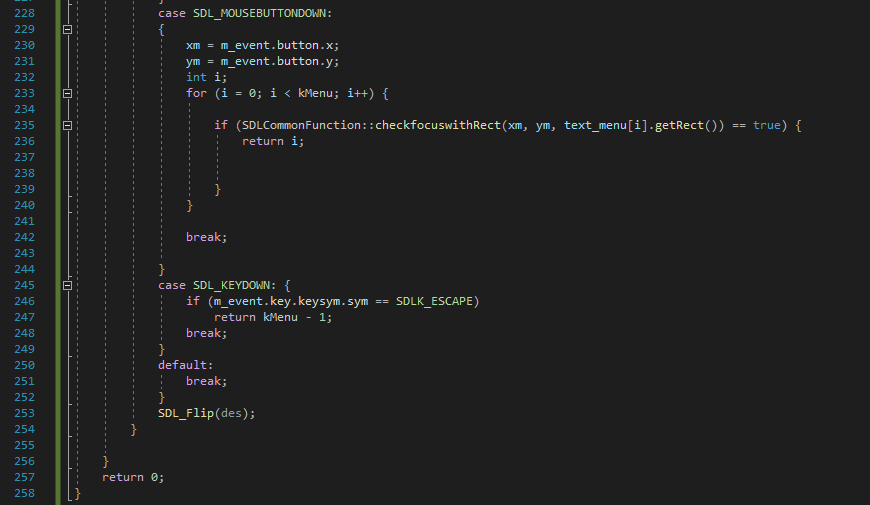
* + - * Sau đó, ta bắt đầu lấy các sự kiện (event) mà người dùng thao tác. Trường hợp SDL\_MOUSEMOTION (di chuyển chuột), ta sẽ thay đổi màu của chữ mỗi khi người dùng di chuột đến vùng có chữ. Dưới đây là code và ví dụ của việc di chuột



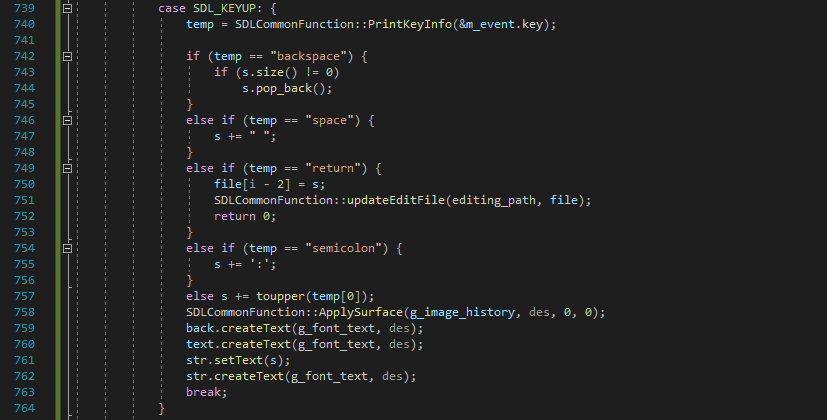




* + - * Tương tự với việc di chuột, ta cũng có việc nhấn chuột cũng là một event. Ta sẽ trả về giá trị mỗi khi các dòng chữ trên màn hình được click vào; lúc đó ta có thể di chuyển đến một cửa sổ mới. Sau khi kết thúc các trường hợp của m\_event, ta sẽ update lại mành hình bằng hàm SDL\_Flip và kết thúc hàm.



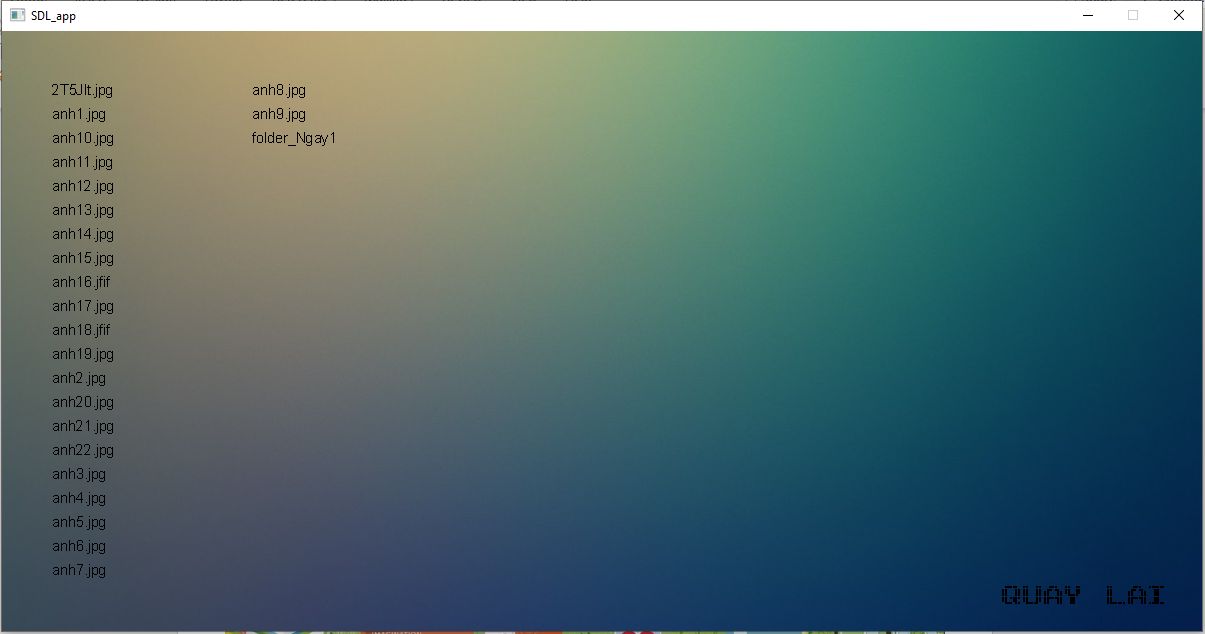
* + Hàm Edit(): tương tự với hàm showMenu(), tuy nhiên ở hàm Edit(), ngoài các sự kiện với chuột, ta còn có các sự kiện với phím. Dưới đây là xử lí sơ bộ cho việc Nhập/Xuất:



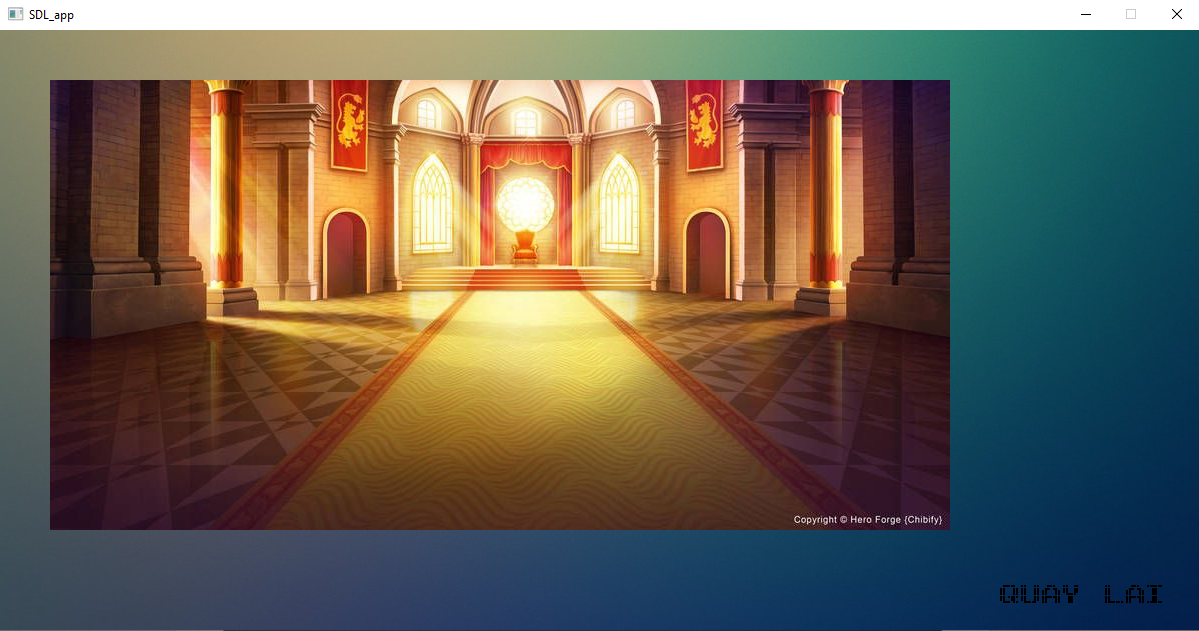
* Dưới đây là màn hình của toàn bộ chương trình:
  + Menu chính



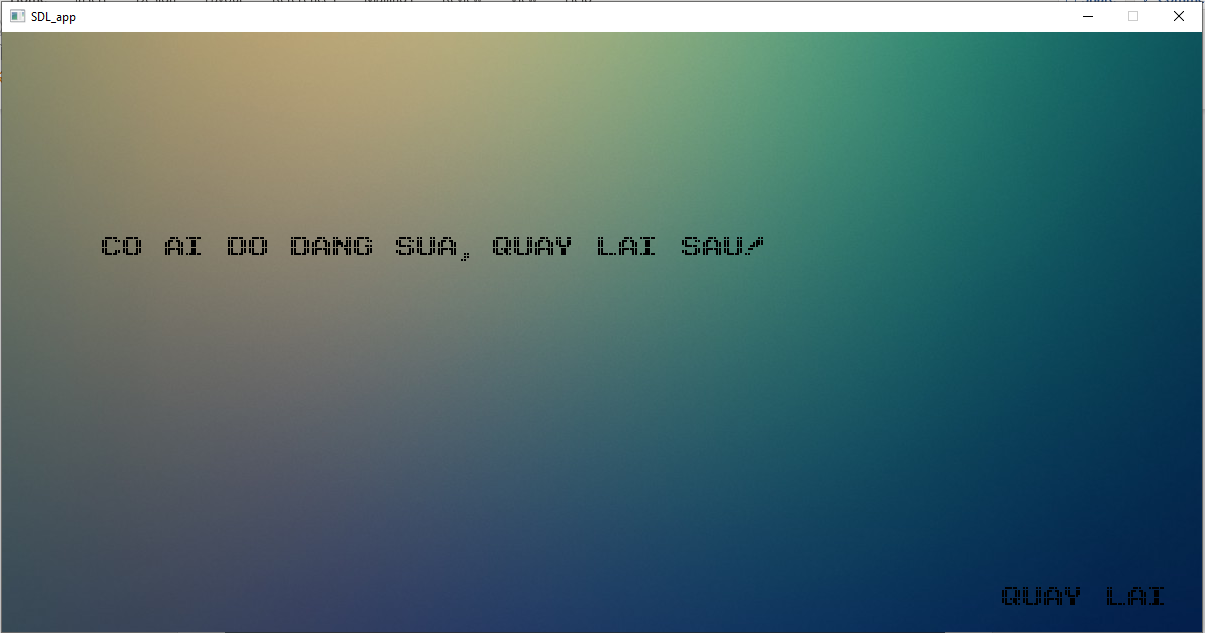
* + Menu xem ảnh



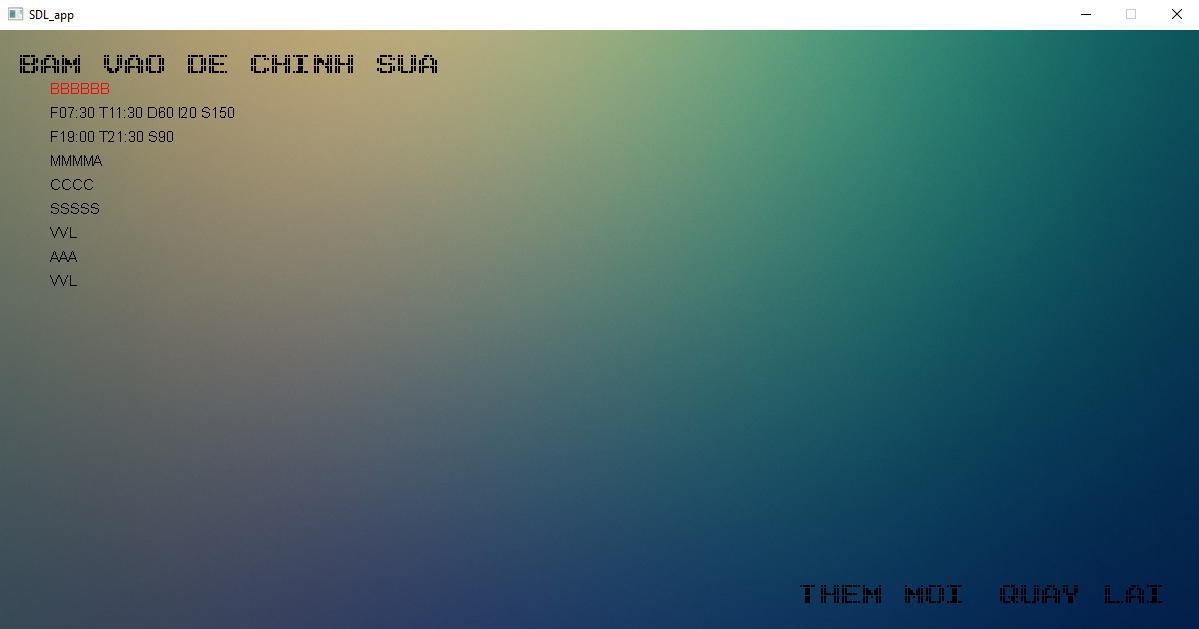
* + Click vào ảnh thì hiển thị ảnh



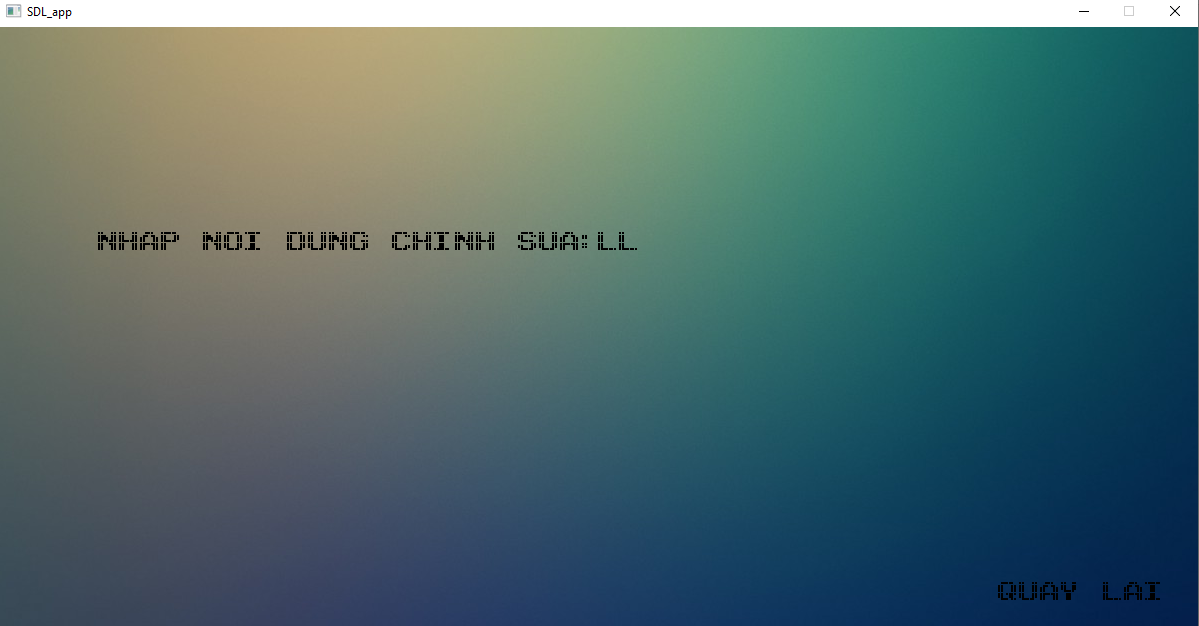
* + Menu chỉnh sửa khi có người đang chỉnh



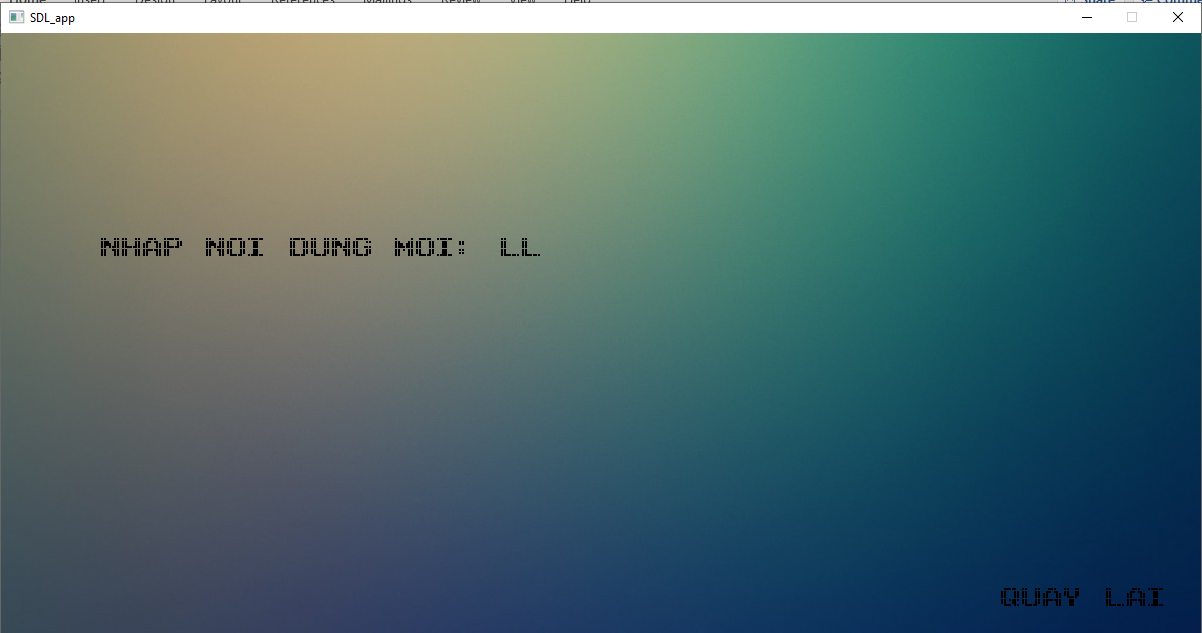
* + Nếu không có ai chỉnh sửa



* + Khi click vào một nội dung

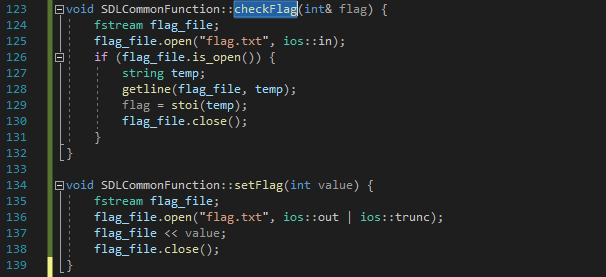


* + Khi click vào “THEM MOI”

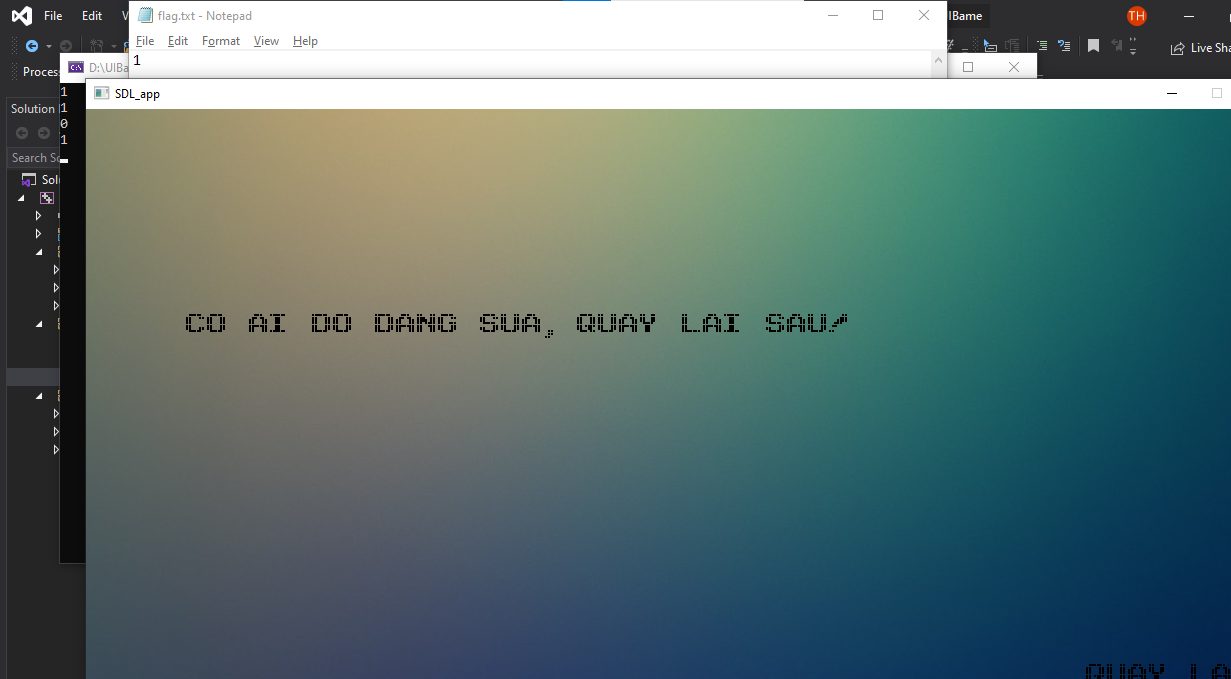


## Về xử lí critical section

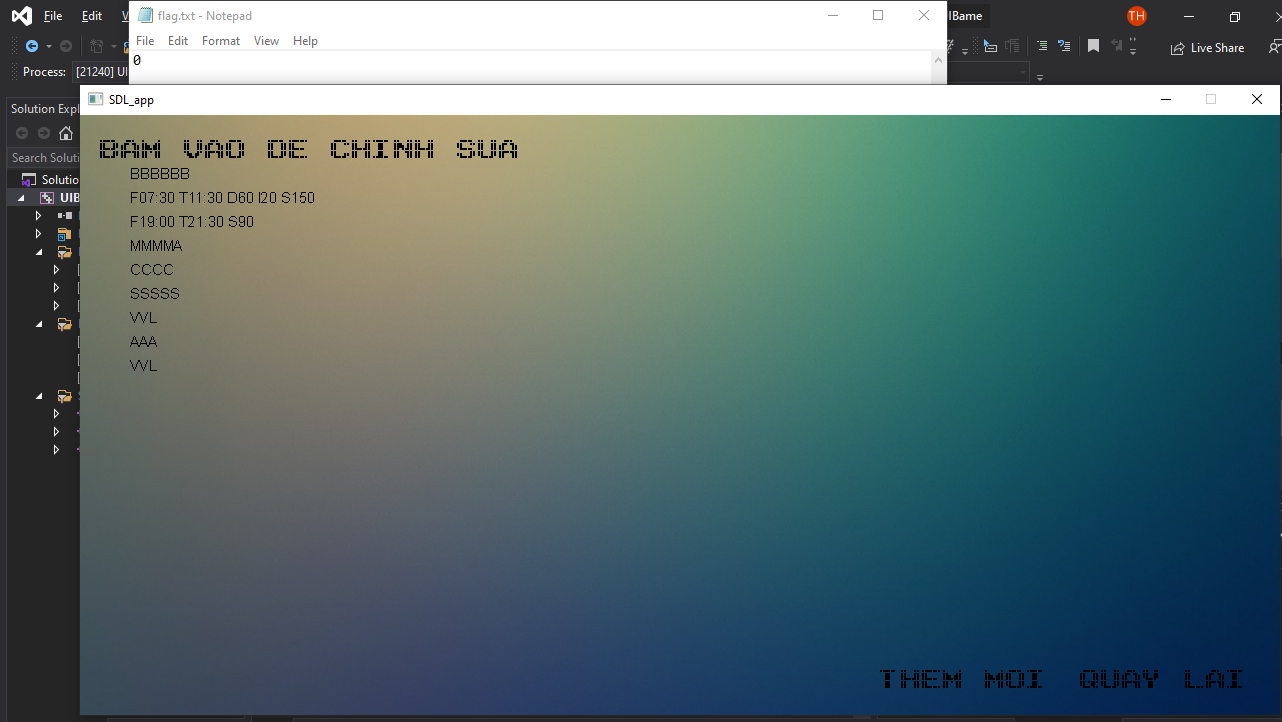
* Ở đây, ta sẽ sử dụng cờ hiệu (flag) để xử lí tranh chấp giữa 2 người dùng. Nếu có người đang chỉnh sửa, flag sẽ lưu thành 1 và với người dùng đến sau, sẽ không có quyền chỉnh sửa. Ngược lại, nếu flag = 0 thì người dùng đến sau sẽ có quyền chỉnh sửa.
* Ta có hàm checkFlag() và setFlag() để lấy và chỉnh sửa lại cờ hiệu trong file:



* Đây là kết quả với việc sử dụng cờ hiệu:
  + Nếu flag = 1:



* + Nếu flag = 0:



# **THAM KHẢO**

*Introduce to SDL*. (n.d.). Retrieved from https://thenumbat.github.io/cpp-course/sdl2/01/01.html

*Lập trình game với SDL*. (n.d.). Retrieved from https://phattrienphanmem123az.com/lap-trinh-game-cpp/bai-1-gioi-thieu-cai-dat.html?fbclid=IwAR38kocA-SFmz3SEi70BSDm\_qIzEElEKdpBrS5N7oVKk5jq\_90bhiCAr7xk

*SDL Input Handling*. (n.d.). Retrieved from https://www.libsdl.org/release/SDL-1.2.15/docs/html/guideinputkeyboard.html