**TRƯỜNG ĐẠI HỌC NHA TRANG**

**KHOA CÔNG NGHỆ THÔNG TIN**



**ĐỌC VÀ PHÁT SINH, MÃ HÓA, GIẢI MÃ QRCODE**

GVHD : Nguyễn Mạnh Cương

SVTH : Nguyễn Chí Hiếu-63134016

Lớp : 63CNTT1

Khánh Hòa, tháng 01 năm 2023

**MỤC LỤC**

[TÓM TẮT 4](#_Toc155646910)

[NỘI DUNG NGUYÊN CỨU 5](#_Toc155646911)

[1 GIỚI THIỆU 6](#_Toc155646912)

[1.1 Mã QR CODE 6](#_Toc155646913)

[1.1.1 Mã QR CODE là gì? 6](#_Toc155646914)

[1.1.2 QR Code khác gì với mã vạch truyền thống (BarCode)? 6](#_Toc155646915)

[1.1.3 Tính thẩm mỹ 7](#_Toc155646916)

[1.1.4 Khả năng lưu trữ 7](#_Toc155646917)

[1.1.5 Ứng dụng của QRcode: 8](#_Toc155646918)

[1.2 Mả Tả Thuật Toán 9](#_Toc155646919)

[2 Phương Pháp Nguyên Cứu 20](#_Toc155646920)

[2.1 Tạo mã QRCODE 20](#_Toc155646921)

[2.2 Đọc QRCcode từ FILE 21](#_Toc155646922)

[2.3 Đọc mã QRCode từ CAMERA 22](#_Toc155646923)

[2.4 Điểm Danh 24](#_Toc155646924)

[2.4.1 Nhận file điểm danh 24](#_Toc155646925)

[2.4.2 Xử lý file điểm danh 25](#_Toc155646926)

[2.5 Chương trình chính 30](#_Toc155646927)

[3 Kết Quả 32](#_Toc155646928)

[TÀI LIỆU THAM KHẢO 35](#_Toc155646929)

[hình 1:1 QR Code 6](#_Toc155646866)

[hình 1:2 QR Code và BarCode 7](#_Toc155646867)

[hình 1:3 Cấu trúc QRCode 9](#_Toc155646868)

[hình 1:4 Mode Indicator 10](#_Toc155646869)

[hình 1:5 Bảng sửa lỗi 13](#_Toc155646870)

[hình 1:6 Hình nhận diện QR Code 16](#_Toc155646871)

[hình 1:7 Bảng hiển thị các Bit thông tin 17](file:///D:\QRCode_TTCS_2023\BCTTCS_NguyenChiHieu_63134016.docx#_Toc155646872)

[hình 2:1 Module tạo mã QRCode 20](#_Toc155646873)

[hình 2:2 Module đọc mã QRCode từ File 21](#_Toc155646874)

[hình 2:3 Module đọc mã QRCode từ Camera 22](#_Toc155646875)

[hình 2:4 Module nhận File điểm danh 24](#_Toc155646876)

[hình 2:5 Hàm tìm cột trống 25](#_Toc155646877)

[hình 2:6 Hàm thêm dữ liệu vào file 27](#_Toc155646878)

[hình 2:7 Hàm xử lý hình ảnh 28](#_Toc155646879)

[hình 2:8 Chương trình chính 31](#_Toc155646880)

[hình 3:1 Giao diện chính 32](#_Toc155646881)

[hình 3:2 Tạo mã QRCode 33](#_Toc155646882)

[hình 3:3 Đọc mã QRCode từ file 33](#_Toc155646883)

[hình 3:4 Đọc mã Qrcode từ camera 34](#_Toc155646884)

[hình 3:5 Thực hiện chọn file điểm danh execl 34](#_Toc155646885)

[hình 3:6 Mẫu định dạng file excel 35](#_Toc155646886)

# TÓM TẮT

Trong phần Giới thiệu đã nêu ra khái niệm về QRCode. Ngoài ra, còn giới thiệu một cách khá tổng quát về các thuật toán tạo, mã hóa QRCode.

Trong phần Phương pháp đã đề cập đến tất cả các chương trình con có trong chương trình chính.

Tiếp theo là phần Kết quả, phần này chỉ ra các kết quả có được khi chạy chương trình với các chức năng

Quy trình thực hiện các bước từ thiết kế và cài đặt thuật toán để tạo ra một chương trình Quét mã QRCode hoàn chỉnh đều được thực hiện bằng python thông qua ứng dụng Visual studio.

Toàn bộ mã nguồn của báo cáo được tải lên theo địa chỉ

<https://github.com/Zinxus/QRCode_TTCS_2023>

# NỘI DUNG NGUYÊN CỨU

Thực hiện việc tạo mã QR Code từ nội dung người dùng nhập vào và hiện mã QR Code, thực hiện lưu mã QR Code nếu người dùng muốn lưu.

Đọc mã QR Code từ file người dùng cung cấp có chứa mã QR Code và hiện nội dung cho người dùng.

Đọc mã QR Code thông qua camera nhận diện mã QR Code và hiện nội dung cho người dùng.

Thực hiện điểm danh sinh viên bằng mã QR Code cá nhân từ file excel người dùng cung cấp với định dạng cho sẵn.

# GIỚI THIỆU

## Mã QR CODE

### Mã QR CODE là gì?

QR Code (mã QR) là viết tắt của **Quick response code** (Tạm dịch: Mã phản hồi nhanh), hoặc có thể gọi là Mã vạch ma trận (Matrix-barcode) hay Mã vạch 2 chiều (2D). Đây là một dạng thông tin được mã hóa để hiển thị sao cho máy có thể đọc được.

QR Code xuất hiện lần đầu tiên vào năm 1994, được tạo ra bởi Denso Wave (công ty con của Toyota). QR Code bao gồm những chấm đen và ô vuông mẫu trên nền trắng, có thể chứa những thông tin như URL, thời gian, địa điểm của sự kiện, mô tả, giới thiệu một sản phẩm nào đó,...

QR Code cho phép quét và đọc mã nhanh hơn bằng các thiết bị như máy đọc mã vạch hoặc [điện thoại](https://www.thegioididong.com/dtdd) có camera với ứng dụng cho phép quét mã, vô cùng tiện lợi cho người dùng.



hình 1:1 QR Code

### QR Code khác gì với mã vạch truyền thống (BarCode)?

Cùng là mã vạch nhưng QR Code lại là phiên bản cải tiến của mã vạch truyền thống. Mã vạch truyền thống là một dãy các vạch được xếp liền kề nhau, chỉ chứa được tối đa **20 ký tự số**, trong khi đó thì mã QR có khả năng chứa tối đa **7.089 ký tự số** và **4.296 ký tự chữ số**.

Điều này cho phép lượng thông tin truyền tải sẽ nhiều hơn, hỗ trợ tốt hơn cho người dùng, đặc biệt là những doanh nghiệp kinh doanh muốn gửi thông điệp đến khách hàng của mình.

### ****Tính thẩm mỹ****

Không chỉ thế, nếu so về kích thước thì QR Code chiếm ít không gian hơn rất nhiều so với mã vạch truyền thống. Nếu in trên sản phẩm hoặc danh thiếp thì sẽ nhỏ gọn và tăng tính thẩm mỹ hơn.



hình 1:2 QR Code và BarCode

### Khả năng lưu trữ

Mặc dù lúc đầu mã QR được dùng để theo dõi các bộ phận trong sản xuất xe hơi, hiện nay nó được dùng trong quản lý kiểm kê ở nhiều ngành khác nhau. Gần đây hơn, phần mềm đọc mã QR đã được cài vào điện thoại di dộng có gắn camera (camera phone) ở Nhật. Điều này đưa đến các ứng dụng mới và đa dạng hướng về người tiêu dùng, nhằm làm đơn giản việc nhập dữ liệu vào điện thoại di động, vốn không mấy hấp dẫn. Mã QR cũng được thêm vào danh thiếp, làm đơn giản đi rất nhiều việc nhập dữ kiện cá nhân của người mới quen vào sổ địa chỉ trên điện thoại di động.

Người dùng có chương trình thu tín hiệu (capture program) và máy tính có giao diện RS­232C có thể dùng máy quét ảnh (scanner) để thu dữ liệu.

Tiêu chuẩn Nhật Bản cho các mã QR, JIS X 0510, được công bố vào tháng giêng năm 1999 và Tiêu chuẩn Quốc tế ISO tương ứng, ISO/IEC18004, được chấp thuận vào tháng sáu năm 2000.

**Khả năng lưu trữ dữ liệu mã QR**

Số đơn thuần Tối đa 7.089 kí tự

Số và chữ cái in Tối đa 4.296 kí tự

Số nhị phân (8 bit) Tối đa 2.953 byte

Kanji/Kana Tối đa 1.817 kí tự

**Khả năng sửa chữa lỗi**

Mức L 7% số từ mã (codeword) có thể được phục hồi.

Mức M 15% số từ mã có thể được phục hồi.

Mức Q 25% số từ mã có thể được phục hồi.

Mức H 30% số từ mã có thể được phục hồi

### Ứng dụng của QRcode:

QR Code ban đầu được phục vụ cho công việc sản xuất ô tô với khả năng mang nhiều dữ liệu mã hóa hơn, độ dung lỗi cao hơn và thời gian quét nhanh hơn. Sau đó, khi các ứng dụng sử dụng QR Code trên điện thoại thông minh (smart phone) xuất hiện thì QR Code dần được sử dụng rộng rãi hơn ở nhiều lĩnh vực. Một số ứng dụng của mã QR ở Nhật Bản và châu Á gồm: ứng dụng của mã QR trong điện thoại di động; thiết bị chữa trị và chuẩn đoán bệnh trong y học; mã vạch QR và công nghệ nhận dạng bằng tần số sóng radio (RFID) ứng dụng trong chuỗi cung ứng…

Trong tương lai, mã vạch QRcode có thể được sử dụng trong các lĩnh vực như:

* Sử dụng tại các bến xe bus : người sử dụng khi quét mã QR Code của bến xe bus sẽ biết thông tin về các chuyến xe sắp tới.
* Sử dụng tại bảo tàng : người sử dụng chỉ cần quét mã QR Code đặt cạnh vật trưng bày là biết được thông tin chi tiết và cập nhật về đồ vật đó.
* Sử dụng để mua hàng ở bất kỳ đâu : người sử dụng khi đi tàu điện ngầm, xe bus … nếu thấy thích mặt hàng đang quảng cáo trên đó có thể đặt mua ngay lập tức thông qua QR Code và Mobile Internet.
* Sử dụng tại siêu thị : người mua có thể quét mã QR Code để biết được hàm lượng dinh dưỡng của đồ ăn cần mua.
* Sử dụng tại các hội thảo : người tham gia hội thảo có thể sử dụng QR Code thay cho Business Card của mình. Sử dụng với các tờ báo/tạp chí giấy : người đọc có thể quét mã QR Code được in trong tờ báo/tạp chí giấy để truy cập phiên bản online/mobile của tờ báo/tạp chí này.
* Sử dụng tại quán bar/club : để xác định xem ca sỹ, ban nhạc, bài nhạc đang chơi là ai/bài nhạc gì?
* Sử dụng với các món ăn : để biết được công thức và cách chế biến món ăn.
* Sử dụng với đồ vật cá nhân (xe, áo thun…) : để cung cấp thông tin về chủ nhân…
* Sử dụng để thu tiền xe ô tô: thay vì việc phải dừng lại ở trạm thu phí, xe ô tô có thể di chuyển qua trạm với tốc độ vừa phải mà vẫn nộp được phí nếu được gắn QR Code trên xe.

## Mả Tả Thuật Toán

hình 1:3 Cấu trúc QRCode

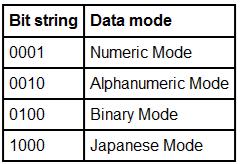
**Tạo một chuỗi nhị phân:**

Bước đầu tiên để tạo một mã QR code là tạo một chuỗi nhị phân bao gồm dữ liệu và thông tin về chế độ mã hóa, cũng như độ dài của dữ liệu.

Trong phần hướng dẫn này tôi sẽ hướng dẫn mã hóa chuỗi HELLO WORLD ở QR code phiên bản 1, với mức độ sửa lỗi là Q.

**Bước 1:** Mã hóa Bộ chỉ chế độ (Mode Indicator)

Một bộ chỉ chế độ là một chuỗi 4 bit thể hiện chế độ dữ liệu bạn đang sử dụng: kiểu số, kiểu chuỗi, kiểu nhị phân hoặc Nhật ngữ.



hình 1:4 Mode Indicator

Với chuỗi HELLO WORLD, là kiểu chuỗi, vậy nên Bộ chỉ chế độ sẽ là 0010.

Chuỗi nhị phân: 0010

**Bước 2:** Mã hóa độ dài của dữ liệu

Trong bước này, chúng ta xác định có bao nhiêu ký tự trong thông điệp, và chuyển độ dài đó thành một số nhị phân.

Với chuỗi HELLO WORLD, có 11 ký tự kể cả khoảng trắng. Ta chuyển 11 sang nhị phân, được 1011.

Xem danh sách ở dưới. Khi mã hóa độ dài của dữ liệu, chúng ta mã hóa nó sử dụng một số lượng bit đặc tả. Như đã nói ở đầu, chúng ta sử dụng QR code phiên bản 1, vậy nên chúng ta cần sử dụng 9 bit để mã hóa độ dài dữ liệu. Ta thêm số 0 vào đầu chuỗi cho đên khi đủ 9 bit: 000001011.

**Phiên bản 1 tới 9**

* Kiểu số: 10 bits
* Kiểu chuỗi: 9 bits
* Kiểu nhị phân: 8 bits
* Kiểu Nhật ngữ: 8 bits

**Phiên bản 10 tới 26**

* Kiểu số: 12 bits
* Kiểu chuỗi: 11 bits
* Kiểu nhị phân: 16
* Kiểu Nhật ngữ: 10 bits

**Phiên bản 27 tới 40**

* Kiểu số: 14 bits
* Kiểu chuỗi: 13 bits
* Kiểu nhị phân: 16 bits
* Kiểu Nhật ngữ: 12 bits
* Tới đây chuỗi nhị phân chúng ta là: 0010 000001011.

**Bước 3:** Mã hóa dữ liệu

Để mã hóa chuỗi ký tự, ta chia chuỗi thành các cặp ký tự: HE,LL,O ,WO,RL,D. Với mỗi cặp ký tự, ta lấy giá trị mã ASCII của ký tự đầu nhân với 45, sau đó cộng với giá trị mã ASCII của ký tự sau. Sau đó chuyển kết quả thu được thành một chuỗi nhị phân 11 bit.

Đối với các ký tự không có cặp, tao lấy giá trị mã ASCII của nó rồi chuyển thành chuỗi nhị phân 6 bit.

Đối với các chế độ khác: kiểu số, kiểu nhị phân, kiểu Nhật ngữ, tao sử dụng phương thức khác để mã hóa dữ liệu.

Kiểu dữ liệu hóa dữ liệu

**Bước 4:** Hoàn thành các bit

Với chuỗi bit thu được ở trên, ta phải chắc chắn rằng nó có đủ độ dài. Điều này phụ thuộc vào việc có bao nhiêu bit chúng ra cần để tạo phiên bản và sửa lỗi.

Với QR code phiên bản 1, sửa lỗi mức độ Q. Chúng ta phải tạo một chuỗi 104 bit. Tham khảo thêm: version capacity table

(<http://www.denso­wave.com/qrcode/vertable1­e.html>) . Nếu chuỗi bit nhỏ ít hơn 104, ta phải thêm 4 số 0 vào cuối chuỗi. Nếu thêm 4 số 0 được một chuỗi nhiều hơn 104, ta chỉ cần thêm số 0 cho tới khi nào đủ 104 bit.

Chuỗi của chúng ta có 59 bit, ta thêm 4 số 0 vào cuối chuỗi. (Còn nếu chuỗi có 102 bit thì ta thêm 2 số 0).

Chuỗi của ta bây giờ là:

0010 000001011 01100001011 01111000110 10001011100 10110111000 10011010100 001101 **0000**

**Bước 5:** Giới hạn chuỗi thành các chuỗi con 8 bit, sau đó thêm các số 0 nếu cần.

Bước này ta chia chuỗi thành các nhóm 8 bit.

Ở chuỗi con cuối cùng, nếu chưa đủ 8 bit thì ta thêm vào sau đó các số 0 cho tới khi đủ. Chuỗi của chúng ta thêm 2 số 0 vào sau chuỗi con cuối cùng.

00100000 01011011 00001011 01111000 11010001 01110010 11011100 01001101 01000011 010000**00**  
**Bước 6: Thêm các từ vào cuối nếu chuỗi quá ngắn**

**Nếu chuỗi bit của chúng ta vẫn chưa đủ dài, có hai chuỗi con đặc biệt:**11101100 và 00010001. Ta thêm thay phiên hai chuỗi con này vào chuối chuỗi của chúng ta.

Chuỗi của chúng ta chỉ có 10 chuỗi con, cần phải thêm vào 3 chuỗi con nữa. Bắt đầu thêm 11101100, sau đó thêm 00010001, rồi tới 11101100. Nếu cần nhiều hơn thì cứ thêm lần lượt như vậy cho tới khi đủ số chuỗi con.

Chuỗi của chúng ta bây giờ:

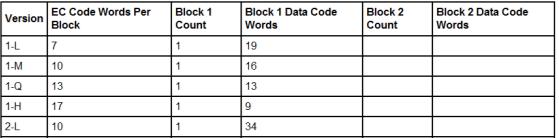
00100000 01011011 00001011 01111000 11010001 01110010 11011100 01001101 01000011 01000000 **11101100 00010001 11101100**

**Tạo bộ mã sửa lỗi**

QR code bao gồm bộ mã sửa lỗi. Các khối dữ liệu dự phòng đảm bảo rằng mã QR code vẫn được đọc cho dù có một phần không đọc được. Tạo mã này rất phức tạp, vì thế nên sau đây sẽ hướng dẫn một cách đơn giản.

QR code sử dụng bộ sửa lỗi Reed­Solomon.

**Bước 1:** Tìm ra có bao nhiêu mã sửa lỗi cần được tạo

Bước đầu tiên để tạo mã sửa lỗi là xác định có bao nhiêu từ để tạo cho phiên bản QR và mức độ sửa lỗi. Xem bảng sửa lỗi (http://www.thonky.com/qr­code­tutorial/error­correction­table/) để biết thêm thông tin chi tiết

hình 1:5 Bảng sửa lỗi

Xem bảng ở trên, chúng ta cần 13 mã sửa lỗi.

**Bước 2:** Tạo một thông điệp đa thức

Chúng ta chuyển 13 khối dữ liệu ở trên thành dạng thập phân:

00100000 01011011 00001011 01111000 11010001 01110010 11011100 01001101 01000011 01000000 11101100 00010001 11101100

Ta được:

32, 91, 11, 120, 209, 114, 220, 77, 67, 64, 236, 17, 236

Các số này sẽ là hệ số của đa thức. Ta có:

***32x25 + 91x24 + 11x23 + 120x22 + 209x21 + 114x20 + 220x19 + 77x18 + 67x17 + 64x16 + 236x15 + 17x14 +236x13***

Lũy thừa của số hạng đầu tiên được tính:

Số khối dữ liệu + số mã sửa lỗi ­1

Ở đây là 13+13­1=25. Vậy nên số hạng đầu tiên là ***32x25 .***

**Bước 3:** Tạo bộ tạo đa thức

Tiếp theo ta sẽ tạo một bộ tạo đa thức. Ta chia thông điệp đa thức bởi bộ tạo này để tạo mã sửa lỗi.

Các bộ tạo đa thức đến từ một cái gì đó gọi là một trường hữu hạn , cũng được biết đến như là một trường Galois. Các mã QR sử dụng một trường Galois có 256 yếu tố, có nghĩa là, mục đích của chúng ta, rằng những con số mà chúng ta sẽ được giao dịch với sẽ luôn luôn tối đa là 255 và ít nhất là 0.

Để thực hiện việc phân chia đa thức, chúng ta sẽ được chuyển đổi qua lại giữa các ký hiệu α (alpha) và các số nguyên. Alpha và các giá trị số nguyên từ bảng log và antilog.

Bộ tạo đa thức luôn luôn có dạng ***(x – α) (x – α 2) … (x – α t), , t*** tương đương với số lượng mã sửa lỗi cần thiết trừ đi 1.Chúng ta cần 13 mã sửa lỗi, do đó, t trong trường hợp này là 12.

Ta thức hiện triển khai tất cả các thành phần (x – α) cho tới khi chúng ta được đa thức. Công việc này thực hiện bằng tay khá phức tạp, sử dụng generator polynomial tool (http://www.thonky.com/qr­code­tutorial/generator­polynomial­tool/). Ở đây chúng ta tạo 13 mã sửa lỗi:

α0x25 + α74x24 + α152x23 + α176x22 + α100x21 + α86x20 + α100x19 + α106x18 + α104x17 + α130x16 + α218x15 + α206x14 + α140x13+ α78x12

Dưới đây, tôi cho thấy từng bước của việc phân chia đa thức cần thiết để tạo ra mã sửa lỗi. Nếu bạn muốn xem các bước này được thực hiện cho một đa thức tin nhắn khác nhau và số lượng từ ECC, truy cập trang [show division steps (http://www.thonky.com/qr­code­tutorial/show­division­steps/)](http://www.thonky.com/qr-code-tutorial/show-division-steps/).

Kết quả của bước này ta được mã sửa lỗi:

168 72 22 82 217 54 156 0 46 15 180 122 16

Ta đặt đoạn mã sửa lỗi sau đoạn mã dữ liệu:

32 91 11 120 209 114 220 77 67 64 236 17 236 **168 72 22 82 217 54 156 0 46 15 180 122 16**

**Chuyển qua nhị phân:**

00100000 01011011 00001011 01111000 11010001 01110010 11011100 01001101 01000011 01000000 11101100 00010001 11101100 10101000 01001000 00010110 01010010 11011001 00110110 10011100 00000000 00101110 00001111 10110100 01111010 00010000

**Chọn mẫu mặt nạ tốt nhất**

Bây giờ chúng ta đã mã hóa các dữ liệu, tất cả những gì còn lại là chọn mẫu mặt nạ tốt nhất. Một mẫu mặt nạ thay đổi theo các bit, theo một quy tắc cụ thể. Các mẫu mặt nạ được định nghĩa trong tiêu chuẩn mã QR.

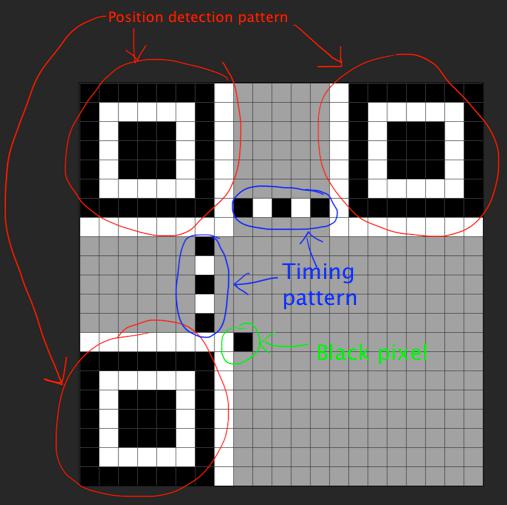
[Có 8 mẫu mặt nạ. Xem danh sách các kiểu mặt nạ QR (http://www.thonky.com/qr­code­tutorial/mask­](http://www.thonky.com/qr-code-tutorial/mask-patterns/) [patterns/) . Sử dụng mô hình mặt nạ cho phép chúng tôi để tạo ra tám mã QR khác nhau và sau đó](http://www.thonky.com/qr-code-tutorial/mask-patterns/) chọn một trong đó sẽ được dễ dàng nhất cho một đầu đọc QR để quét. Một đặc biệt mã QR có thể có các mẫu nhất định hoặc những đặc điểm mà làm cho nó khó khăn cho thiết bị đọc QR chính xác quét mã QR. Ví dụ, nếu điểm ảnh của cùng một màu sắc xảy ra gần nhau, một đầu đọc QR có thể gặp khó khăn khi đọc chính xác các điểm ảnh.

Mỗi mô hình mặt nạ tạo ra một mã QR code khác nhau. Sau khi chúng ta tạo ra các mã khác nhau QR trong nội bộ, chúng tôi cung cấp cho mỗi một số điểm xử phạt theo các quy tắc quy định trong tiêu chuẩn mã QR. Sau đó, chúng tôi cho ra mã QR có số điểm tốt nhất.

**Bước 1: Tạo mã QR code:**

Với QR code phiên bản 1, ta cần một ma trận 21×21.

Tất cả các mã QR code đều có một vài mẫu nhận diện chắc chắc phải đc thể hiện. Ở hình bên dưới, có ba vị trí nhận diện được đặt ở trên cùng bên trái, trên cùng bên phải và dưới cùng bên trái của ma trận. Các mã QR code còn có một pixel đen ở bên phải của vị trí nhận diện bên dưới cùng bên trái. Ngoài ra còn có mẫu timing.



hình 1:6 Hình nhận diện QR Code

**Vị trí điều chỉnh mô hình**

Từ ví dụ này là chỉ mã hóa một phiên bản 1 mã QR, chúng tôi không cần thêm bất kỳ mô hình điều chỉnh vị trí. Tuy nhiên, nếu chúng ta đã mã hóa một mã QR phiên bản 2 hoặc lớn hơn, chúng tôi sẽ thêm các mẫu điều chỉnh vị trí ma trận. Bảng dưới đây liệt kê các tọa độ của nơi để đặt các mô hình điều chỉnh vị trí. Xem ví dụ ở bảng dưới đây.

Bảng liệt kê tọa độ của nơi để đặt các mô hình điều chỉnh vị trí

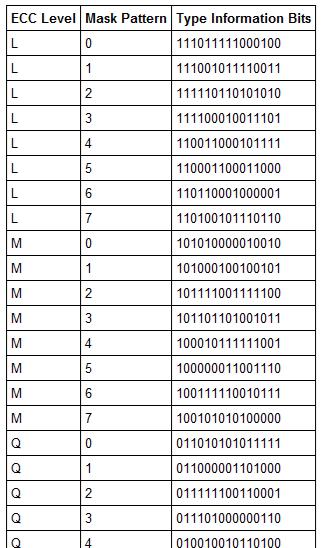
Ví dụ, hình ảnh dưới đây cho thấy các mô hình điều chỉnh vị trí cho phiên bản mã QR 8. Mỗi mô hình điều chỉnh vị trí là một điểm ảnh màu đen bao quanh bởi các điểm ảnh màu trắng được bao quanh bởi các điểm ảnh màu đen. Theo bảng trên, cho phiên bản mã QR 8, vị trí các mô hình điều chỉnh 6, 24, và 42. Như ta thấy trong hình ảnh, mô hình điều chỉnh vị trí được đặt ở tất cả các giao điểm của ba con số này, ngoại trừ đã có lớn hơn mô hình phát hiện vị trí **không đặt mô hình điều chỉnh vị trí trên hàng đầu của các mô hình phát hiện vị** trí trong **góc của mã QR.**

Mô hình điều chỉnh vị trí

**Thêm thông tin Loại**

Các thông tin về mức độ sửa lỗi và mô hình mặt nạ được mã hóa ở dạng dải với các cạnh của các mẫu phát hiện vị trí.

Bảng sau đây hiển thị các bit thông tin các loại được yêu cầu cho các cấp, sửa lỗi khác nhau và các mẫu mặt nạ.



hình 1:7 Bảng hiển thị các Bit thông tin

Hình hiển thị các Bit thông tin

Trong ví dụ Hello World của chúng tôi, mô hình mặt nạ tốt nhất là 0 (điều này được giải thích dưới đây). Chúng ta đang sử dụng mức độ hiệu chỉnh lỗi Q, vì vậy các chuỗi loại thông tin là 011010101011111. Như trong hình trên, các chuỗi loại thông tin được hiển thị hai lần. Một trong số đó là dưới hai mô hình phát hiện vị trí hàng đầu, bắt đầu từ bên trái. Nó bỏ qua quá khứ mô hình thời gian và phần dữ liệu của lưới. Điều thứ hai là quyền của các mẫu phát hiện vị trí, bắt đầu từ phía dưới, bỏ qua dấu chấm đen, phần dữ liệu, và mô hình thời gian. Cả hai sọc cùng một chuỗi: 011010101011111.

**Thêm thông tin phiên bản**

Nếu mã QR của bạn là phiên bản 7 hoặc lớn hơn, bạn cần phải thêm các bit thông tin phiên bản ma trận. Đây là những được đặt ở bên trái của mô hình trên bên phải phát hiện vị trí và trên các mô hình phát hiện vị trí phía dưới bên trái, như thể hiện trong hình ảnh dưới đây, trong đó có một phiên bản 9 Mã QR.

Hình thêm thông tin phiên bản QR Code

Bảng sau đây cho thấy một danh sách của các bit thông tin phiên bản cần thiết cho mỗi phiên bản. Những bit được đặt trong cấu hình sau đây trong các vị trí được chỉ định trong hình trên:

Bảng danh sách các Bit thông tin phiên bản

Trong ví dụ của chúng ta, mã QR là nhỏ hơn so với phiên bản 7, vì vậy chúng ta không cần thêm các bit thông tin phiên bản mã.

Bảng Bit tương ứng từng phiên bản

**Thêm dữ liệu Bits**

Cuối cùng, chúng ta thêm các bit dữ liệu. Trong bước này, chúng ta thay đổi chút tùy thuộc vào mô hình mặt nạ mà chúng ta đang sử dụng.Trước khi thêm chút thực tế, chúng tôi sử dụng các mẫu mặt nạ để quyết định xem có nên thay đổi màu sắc của các bit hiện tại hay không.

**Sắp xếp theo cột từ dưới lên**

Các bit dữ liệu được thêm vào theo một thứ tự cụ thể. Các điểm ảnh đầu tiên được thêm vào phía dưới bên phải của mã QR. Điều thứ hai được đặt ở bên trái đó. Hai tiếp theo được đặt trên đầu những người đầu tiên theo thứ tự. Hai cột điểm ảnh này vẫn tiếp tục đi lên theo cách này, bỏ qua bất kỳ điểm ảnh đã được thiết lập, cho đến khi nó đạt đến hàng đầu của mã QR.

Hình ảnh sắp xếp theo cột

**Xuống cột**

Một khi nó đạt đến đỉnh, một cột hai­pixel mới bắt đầu, thời gian này sẽ đi xuống.

Hình ảnh xuống cột

Khi bạn đạt đến dưới cùng của lưới, thay đổi hướng một lần nữa, và tiếp tục trong mô hình này giống như đường đi của con rắn.

**Chắc chắn bỏ qua mẫu timing**

Khi bạn thêm pixels, hãy chắc chắn để bỏ qua các mô hình thời gian.

Hình ảnh các mẫu timing

Và các mẫu timing ở chiều dọc:

Hình các mẫu timing ở chiều dọc

**Hoàn tất mã QR code**

Tiếp tục theo cách này cho đến khi tất cả của các điểm ảnh dữ liệu đã được đặt. Sau đó, các mã QR là hoàn tất. Tuy nhiên, các đặc điểm kỹ thuật mã QR yêu cầu bạn sử dụng các mô hình mặt nạ tốt nhất có thể

Bước 3: Cung cấp mỗi QR code trong 8 mã một số điểm và chọn mã có số điểm tốt nhất

Trước khi chúng ta có thể cho ra mã QR của chúng ta, chúng ta cần thử lần lượt 8 loại của các mẫu mặt nạ để tìm ra một penalty thấp nhất, dựa trên các quy tắc penalty quy định tại các đặc điểm kỹ thuật mã QR.

Có bốn quy tắc xử phạt:

* Quy tắc đầu tiên cung cấp cho các mã QR một hình phạt đối với từng nhóm năm hoặc nhiều hơn điểm ảnh cùng màu trong một hàng.
* Nguyên tắc thứ hai cung cấp cho các mã QR một hình phạt đối với từng khu vực 2×2 điểm ảnh cùng màu mà nó tìm thấy.
* Nguyên tắc thứ ba cung cấp cho các mã QR một hình phạt lớn nếu có mô hình trông tương tự như các mô hình phát hiện vị trí.

Nguyên tắc thứ tư cung cấp cho các mã QR một hình phạt nếu có nhiều hơn một nửa trong số các điểm ảnh tối hoặc ánh sáng

# Phương Pháp Nguyên Cứu

## Cài đặt môi trường Python và thư viện

Cài đặt môi trường python:

Bước 1: Truy cập vào trang chủ của python

<https://www.python.org/>

Bước 2: Chọn phiên bản ở mục Downloads và chọn phiên bản mới nhất để download.

Bước 3:Sau khi download hoàn tất. Chúng ta nhấn chọn chạy file đã tải về để tiến hành cài đặt

Bước 4: Tick vào ô Add Python 3.12.1 (phiên bản mới nhất)to PATH và chọn Install Now

Bước 5: Khi cửa sổ hiển thị Setup was successful là ta đã cài đặt thành công môi trường Python > Close

Cài đặt các thư viện:

Ta mở Command Prompt (CMD).

* Để mở Command Prompt bạn dùng tổ hợp phím Windows + R để mở hộp thoại Run
* Sau đó, gõ cmd > Enter để mở Command Prompt

Trong cửa sổ Command Prompt, bạn gõ python > Enter để kiểm tra xem đã cài môi trường chưa và thực hiện cài thư viện .

Các thư viện thường dùng ‘**qrcode’,** ‘**pyzbar’, ‘opencv’,** ‘**tkinter’, ‘openpyxl’,…**

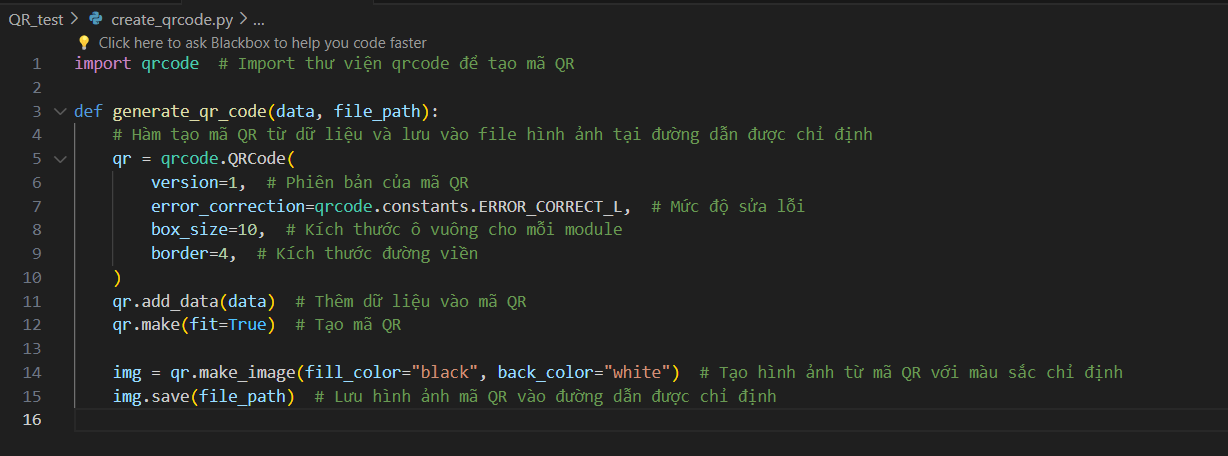
**VD Cách cài đặt:**

Thêm thư viện ‘**qrcode’**. Trong cửa sổ Command Prompt, bạn gõ python –m pip install **qrcode** .

Thêm thư viện ‘**tkinter’.** Trong cửa sổ Command Prompt, bạn gõ python –m pip install **tk** .

Thêm thư viện  **‘opencv’.** Trong cửa sổ Command Prompt, bạn gõ python –m pip install **opencv**.

## Tạo mã QRCODE



hình 2:1 Module tạo mã QRCode

Đầu tiên, thư viện ‘**qrcode’** được ‘**import’** để tạo mã QR. Hàm ‘**generate\_qr\_code’** nhận hai đối số: ‘**data’** là dữ liệu bạn muốn chứa trong mã QR và ‘**file\_path’** là đường dẫn để lưu trữ hình ảnh mã QR.

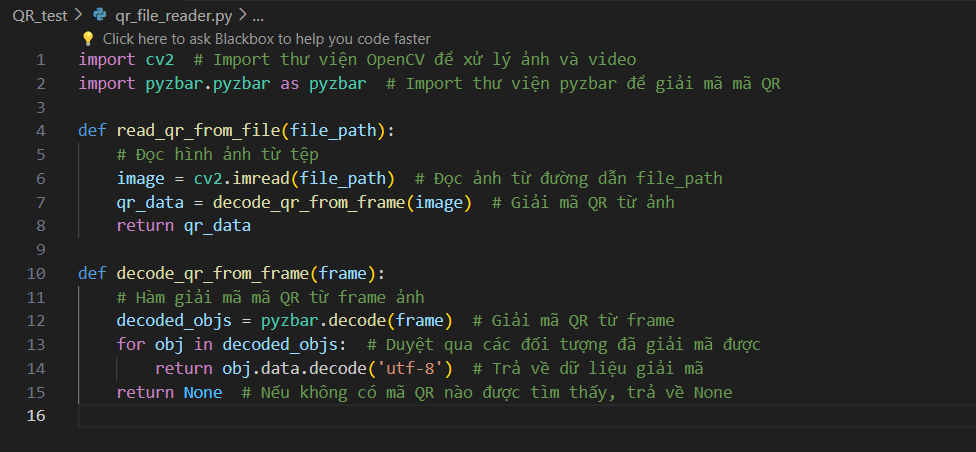
Mã tiếp theo tạo một đối tượng QRCode từ thư viện ‘**qrcode’** với các thông số nhất định như phiên bản của mã QR, mức độ sửa lỗi, kích thước ô vuông cho mỗi module và kích thước đường viền.

Sau đó, dữ liệu được thêm vào mã QR thông qua ‘**qr.add\_data(data)’** và mã QR được tạo ra thông qua ‘**qr.make(fit=True)’.**

Tiếp theo, hình ảnh mã QR được tạo ra từ mã QR đã được tạo trước đó bằng cách sử dụng ‘**qr.make\_image(fill\_color="black", back\_color="white")’**, trong đó màu sắc của module và nền có thể được chỉ định.

Cuối cùng, hình ảnh mã QR được lưu vào đường dẫn được chỉ định thông qua ‘**img.save(file\_path)’.**

## Đọc QRCcode từ FILE



hình 2:2 Module đọc mã QRCode từ File

Đoạn code này sử dụng thư viện OpenCV (**cv2**) để đọc và xử lý hình ảnh và video. Nó cũng sử dụng thư viện ‘**pyzbar’** để giải mã mã QR từ hình ảnh.

Hàm ‘**read\_qr\_from\_file(file\_path)’** nhận đường dẫn đến một tệp hình ảnh, sau đó sử dụng OpenCV để đọc hình ảnh từ đường dẫn này. Sau khi có hình ảnh, nó gọi hàm ‘**decode\_qr\_from\_frame(image)’** để giải mã mã QR từ hình ảnh và trả về dữ liệu QR được giải mã.

Hàm ‘**decode\_qr\_from\_frame(frame)’** nhận một frame ảnh, sử dụng thư viện ‘**pyzbar**’ để giải mã QR từ frame này. Nó lặp qua các đối tượng đã giải mã được (nếu có) và trả về dữ liệu của mã QR đầu tiên được tìm thấy dưới dạng chuỗi UTF-8. Nếu không tìm thấy mã QR nào, hàm sẽ trả về ‘**None’**.

## Đọc mã QRCode từ CAMERA



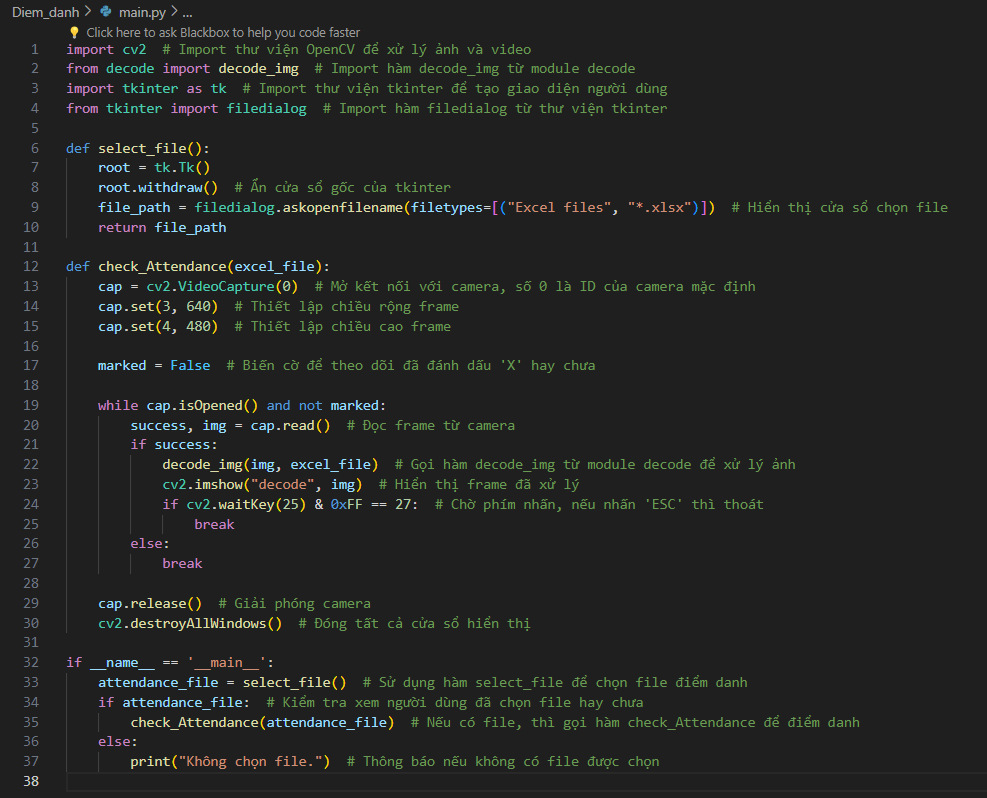
hình 2:3 Module đọc mã QRCode từ Camera

Đoạn code này sử dụng thư viện ‘**threading’** để đọc và xử lý dữ liệu từ camera ở nền và thư viện OpenCV (**cv2**) để đọc và xử lý hình ảnh và video. Nó cũng sử dụng thư viện ‘**pyzbar’** để giải mã mã QR từ hình ảnh. Đoạn mã này là một lớp ‘**QRCameraReader’** được thiết kế để chạy một luồng cảm biến camera và đọc mã QR từ dữ liệu hình ảnh của camera.

1. **‘\_\_init\_\_’**: Được sử dụng để khởi tạo lớp. Nó khởi tạo các biến và có thể nhận một hàm callback để xử lý dữ liệu QR.
2. **‘start\_camera’**: Bắt đầu chạy camera. Nó thiết lập các cờ và mở kết nối với camera. Một luồng riêng biệt được tạo ra để đọc mã QR từ camera.
3. **‘stop\_camera’**: Dừng camera. Nó ngưng việc chạy camera, giải phóng các tài nguyên và đóng tất cả các cửa sổ hiển thị.
4. **‘read\_qr\_from\_camera’**: Đọc mã QR từ camera. Trong một vòng lặp, nó liên tục đọc các frame từ camera, giải mã mã QR từ mỗi frame và hiển thị frame đó trên một cửa sổ gọi là "**QR Scanner**". Nếu nhận diện được mã QR, nó dừng việc chạy camera và gọi ‘**callback function’** để xử lý dữ liệu QR.
5. **‘decode\_qr\_from\_frame’**: Giải mã mã QR từ frame. Sử dụng thư viện ‘**pyzbar’** để giải mã mã QR từ frame hình ảnh. Nếu tìm thấy mã QR, nó trả về dữ liệu QR dưới dạng chuỗi UTF**-**8, nếu không thì trả về None.

## Điểm Danh

### Nhận file điểm danh



hình 2:4 Module nhận File điểm danh

Đoạn code này sử dụng thư viện ‘**tkinter**’để tạo giao diện người dùng và thư viện OpenCV (**cv2**) để đọc và xử lý hình ảnh và video.

**Chọn file Excel:**

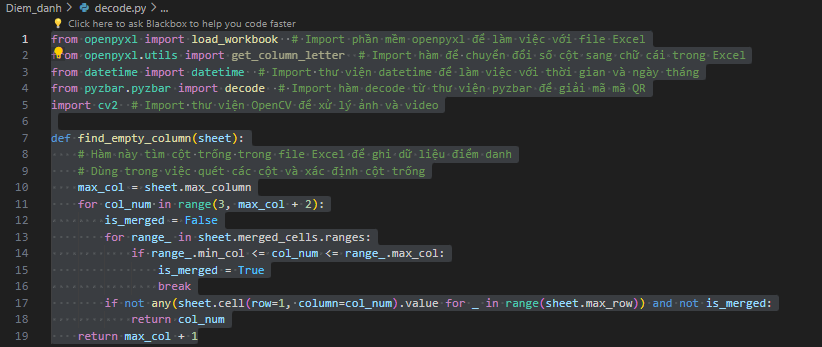
* Hàm ‘**select\_file()**’ sử dụng ‘**tkinter’** để hiển thị cửa sổ cho phép người dùng chọn một file Excel (.xlsx). Người dùng sẽ chọn file này để lưu thông tin điểm danh.
* Sau khi chọn file, nó trả về đường dẫn của file được chọn để sử dụng trong quá trình điểm danh.

**Điểm danh từ camera:**

* Hàm ‘**check\_Attendance(excel\_file)**’ mở kết nối với camera và bắt đầu quá trình điểm danh dựa trên mã QR từ hình ảnh camera.
* Nó sử dụng ‘**cv2.VideoCapture()**’ để kết nối với camera và thiết lập kích thước của frame (chiều rộng và chiều cao).
* Tiếp theo, nó bắt đầu một vòng lặp để liên tục đọc các frame từ camera.
* Mỗi frame được chuyển đến hàm ‘**decode\_img()**’ để giải mã mã QR và điểm danh từ file Excel đã chọn.
* Quá trình điểm danh này diễn ra liên tục cho đến khi người dùng nhấn phím ‘**ESC’** hoặc camera bị đóng.
* Khi kết thúc quá trình điểm danh, chương trình giải phóng kết nối với camera và đóng tất cả các cửa sổ hiển thị.

Tóm lại, chương trình này cung cấp một giao diện để người dùng chọn file Excel và sử dụng camera để quét mã QR để điểm danh dựa trên thông tin từ mã QR được quét từ hình ảnh camera.

### Xử lý file điểm danh



hình 2:5 Hàm tìm cột trống

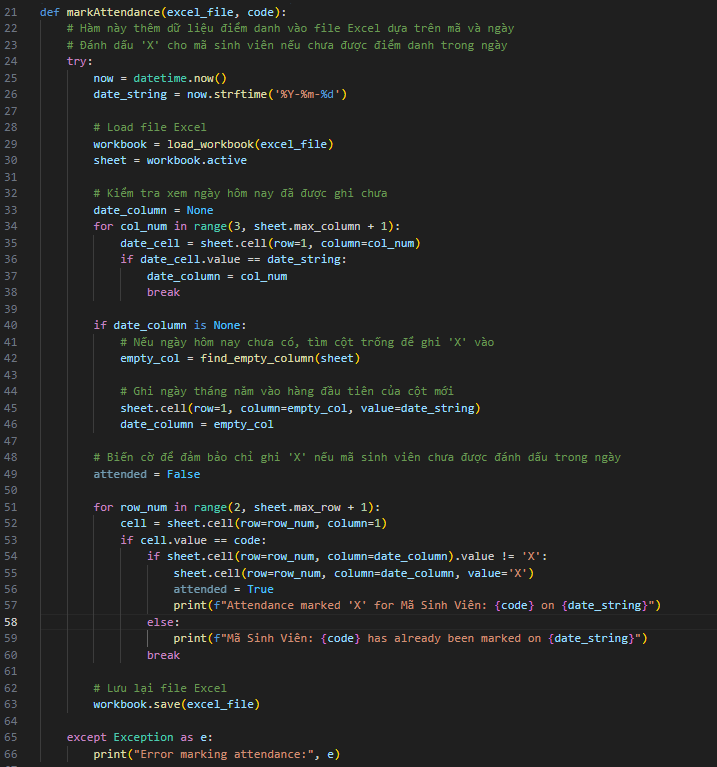
Đoạn code này sử dụng thư viện ‘**openpyxl’** để làm việc với file Excel, thư viện ‘**datatime’** để để làm việc với thời gian và ngày tháng và thư viện OpenCV (**cv2**) để đọc và xử lý hình ảnh và video. Nó cũng sử dụng thư viện ‘**pyzbar’** để giải mã mã QR từ hình ảnh.

Hàm ‘**find\_empty\_column(sheet)**’ trong đoạn code này tìm cột trống trong một bảng tính Excel để có thể ghi dữ liệu điểm danh mới. Đây là cách nó hoạt động:

1. ‘**max\_col = sheet.max\_column**’: Xác định số cột tối đa trong bảng tính.
2. ‘**for col\_num in range(3, max\_col + 2):**’: Duyệt qua các số cột, bắt đầu từ cột thứ 3 đến ‘**max\_col + 2’**.
3. ‘**is\_merged = False**’: Khởi tạo biến cờ để xác định xem cột hiện tại có bị gộp với cột khác hay không.
4. ‘**for range\_ in sheet.merged\_cells.ranges:**’: Duyệt qua tất cả các vùng gộp cột trong bảng tính.
5. ‘**if range\_.min\_col <= col\_num <= range\_.max\_col:**’: Kiểm tra xem cột hiện tại có nằm trong một vùng gộp cột không. Nếu có, cờ ‘**is\_merged**’ sẽ được đặt thành True.
6. ‘**if not any(sheet.cell(row=1, column=col\_num).value for \_ in range(sheet.max\_row)) and not is\_merged:**’:

* Kiểm tra xem cột hiện tại không có giá trị nào trong hàng đầu tiên và không phải là cột đã được gộp.
* ‘**any(sheet.cell(row=1, column=col\_num).value for \_ in range(sheet.max\_row))**’ kiểm tra xem có giá trị nào trong cột không.
* ‘**not is\_merged**’ đảm bảo cột không được gộp với cột khác.

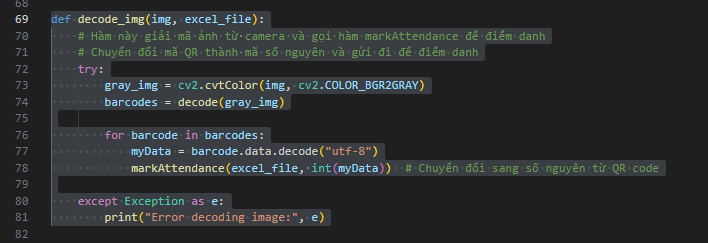
1. Nếu cột thỏa mãn các điều kiện trên, nó sẽ trả về số cột đó (được biểu diễn bằng ‘**col\_num**’).
2. Nếu không tìm thấy cột trống, nó sẽ trả về ‘**max\_col + 1**’, tức là cột tiếp theo sau cột cuối cùng trong bảng tính. Điều này giúp chương trình thêm dữ liệu mới vào cột mới nếu không có cột trống nào



hình 2:6 Hàm thêm dữ liệu vào file

Hàm ‘**markAttendance(excel\_file, code)**’ được sử dụng để thêm dữ liệu điểm danh vào file Excel dựa trên mã và ngày hiện tại. Đây là cách mà hàm này hoạt động:

1. ‘**now = datetime.now() và date\_string = now.strftime('%Y-%m-%d')**’: Lấy ngày hiện tại dưới dạng chuỗi 'YYYY-MM-DD' sử dụng thư viện ‘**datetime’**.
2. ‘**workbook = load\_workbook(excel\_file) và sheet = workbook.active**’: Load file Excel và chọn sheet hiện tại để làm việc.
3. ‘**date\_column = None và for col\_num in range(3, sheet.max\_column + 1):**’: Duyệt qua các cột để kiểm tra xem ngày hiện tại đã được ghi chưa.
4. ‘**if date\_cell.value == date\_string:**’: Nếu ngày hiện tại đã được ghi trong bảng tính, ghi lại cột tương ứng vào biến ‘**date\_column**’.
5. ‘**if date\_column is None: và empty\_col = find\_empty\_column(sheet)**’: Nếu ngày hiện tại chưa được ghi, tìm cột trống để ghi dữ liệu ngày tháng vào.
6. ‘**for row\_num in range(2, sheet.max\_row + 1): và cell = sheet.cell(row=row\_num, column=1)**’: Duyệt qua từng hàng trong cột 1 (chứa mã sinh viên).
7. ‘**if cell.value == code: và if sheet.cell(row=row\_num, column=date\_column).value != 'X':**’: Nếu tìm thấy mã sinh viên và mã này chưa được điểm danh trong ngày, đánh dấu 'X' vào cột ngày tương ứng với mã sinh viên đó.
8. ‘**workbook.save(excel\_file)**’: Lưu lại file Excel sau khi điểm danh.

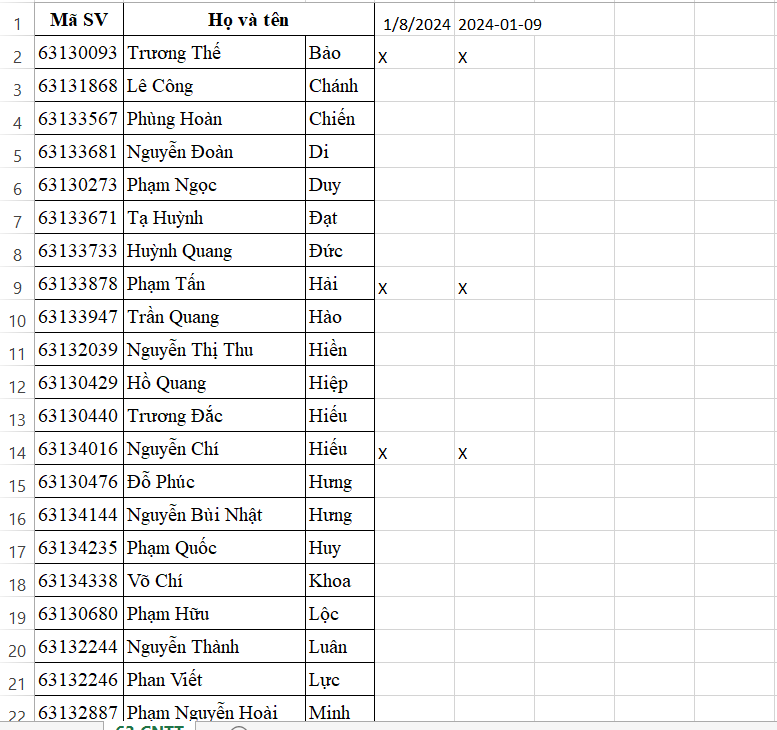


hình 2:7 Hàm xử lý hình ảnh

Hàm `decode\_img` trong đoạn code này nhận một hình ảnh từ camera và giải mã mã QR từ hình ảnh đó, sau đó gọi hàm `markAttendance` để điểm danh người dùng dựa trên dữ liệu mã QR.

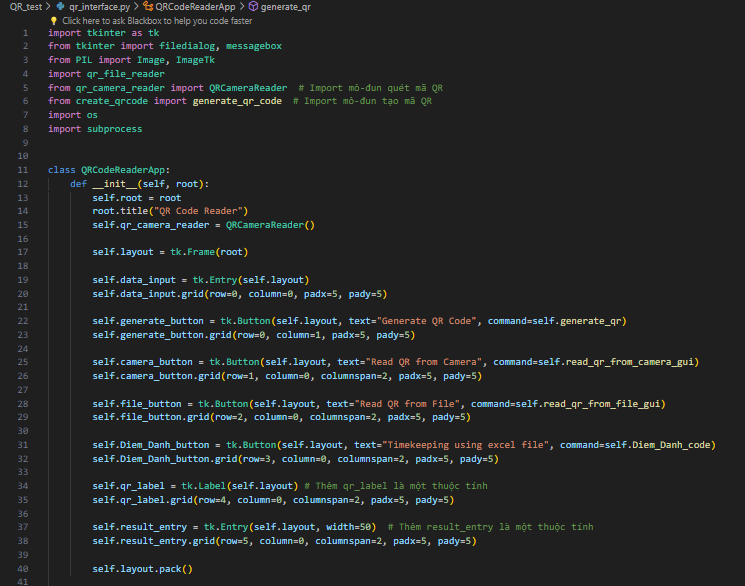
1. ‘**gray\_img = cv2.cvtColor(img, cv2.COLOR\_BGR2GRAY)’**: Chuyển đổi hình ảnh màu sang ảnh xám. Thư viện ‘**pyzbar**’ yêu cầu hình ảnh ở dạng ảnh xám để có thể giải mã mã QR.
2. ‘**barcodes = decode(gray\_img)**’: Sử dụng thư viện ‘**pyzbar’** để giải mã tất cả các mã QR từ hình ảnh xám.
3. ‘**for barcode in barcodes:**’: Duyệt qua danh sách các mã QR được tìm thấy trong hình ảnh.
4. ‘**myData = barcode.data.decode("utf-8")**’: Trích xuất dữ liệu từ mã QR và chuyển đổi nó thành chuỗi UTF-8.
5. ‘**markAttendance(excel\_file, int(myData))**’: Gọi hàm ‘**markAttendance**’ để điểm danh người dùng. Nó chuyển đổi dữ liệu mã QR thành một số nguyên (nếu có thể) trước khi gửi đi để xử lý trong hàm ‘**markAttendance**’.
6. ‘**except Exception as e:**’: Bắt các ngoại lệ có thể xảy ra trong quá trình giải mã hình ảnh và in ra thông báo lỗi nếu có. Điều này giúp người dùng biết nếu có vấn đề xảy ra trong quá trình giải mã mã QR từ hình ảnh.

## File exel điểm danh

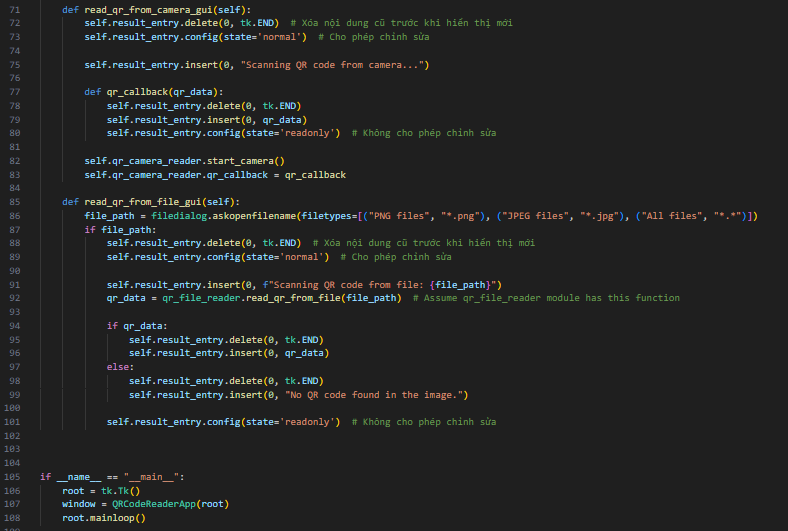


Định dạng file dùng điểm danh sinh viên. Cột đàu tiên chứa Mã SV dùng để so sánh với nội dung mã QR Code cá nhân quét được từ camera.

## Chương trình chính







hình 2:8 Chương trình chính

Đoạn mã này tạo một ứng dụng GUI (Giao diện người dùng đồ họa) để đọc và tạo mã QR thông qua camera hoặc từ file ảnh. Hãy đi từng phần:

**Các thư viện:**

* ‘**tkinter’** được sử dụng để tạo giao diện người dùng.
* ‘**Pillow’** được sử dụng để xử lý các file hình ảnh.
* OpenCV (**cv2**) được sử dụng để đọc và xử lý hình ảnh và video.
* ‘**OS’** được sử dụng để tương tác với hệ điều hành.
* ‘**Subprocess’** được sử dụng để gọi các ứng dụng khác từ Python.

**Giao diện người dùng:**

* Cửa sổ chương trình được tạo và các phần tử như ‘**Entry’**, ‘**Button’**, ‘**Label’** được thêm vào giao diện.

**Chức năng của các nút:**

1. "**Generate QR Code**": Nhập dữ liệu vào ‘**Entry’**, sau đó tạo mã QR từ dữ liệu này. Mã QR được hiển thị trên giao diện và có thể được lưu lại dưới dạng file ảnh PNG nếu người dùng chọn.
2. "**Read QR from Camera**": Bắt đầu quét mã QR từ camera. Khi phát hiện mã QR, nó sẽ hiển thị dữ liệu từ mã QR này lên ‘**Entry’**.
3. "**Read QR from File**": Cho phép người dùng chọn một file ảnh từ đĩa cứng và quét mã QR từ file này. Kết quả quét được hiển thị trên ‘**Entry’**.
4. "**Timekeeping using excel file**": Mở một ứng dụng riêng để thực hiện điểm danh thông qua file Excel. Chức năng này sẽ chạy một file Python khác ("Diem\_danh/main.py").

**Chức năng phụ:**

* **Hiển thị QR Code**: Khi tạo QR Code từ dữ liệu nhập vào, ảnh mã QR được hiển thị trên giao diện.
* **Lưu file QR Code**: Người dùng có thể lưu mã QR dưới dạng file ảnh PNG nếu chọn.

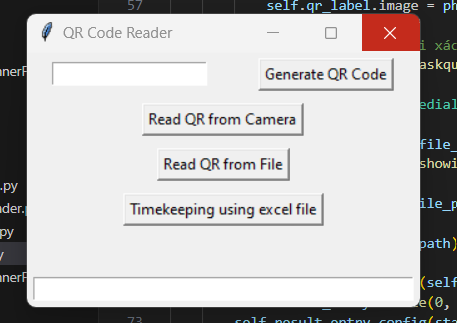
**Điều chỉnh Entry:**

* ‘**result\_entry**’ là nơi hiển thị kết quả sau khi quét hoặc thực hiện các chức năng của ứng dụng. Nó có thể được chỉnh sửa hoặc chỉ đọc tùy thuộc vào tình huống (khi hiển thị kết quả hoặc khi đang quét).

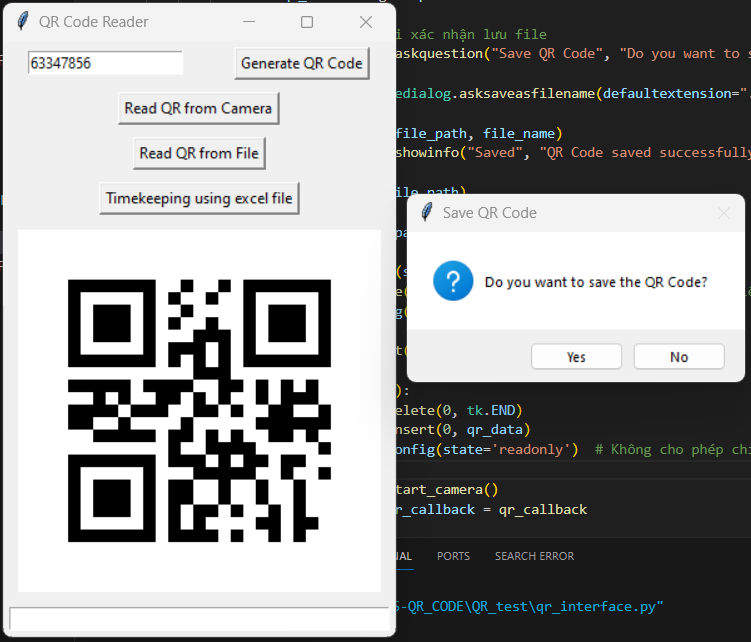
**Quy trình chính:**

* Chương trình bắt đầu từ hàm **‘\_\_main\_\_**’, tạo cửa sổ ứng dụng và gán giao diện người dùng vào cửa sổ này.
* Chương trình chạy vô hạn (vòng lặp ‘**root.mainloop()**’) để duy trì giao diện và chờ sự tương tác từ người dùng.

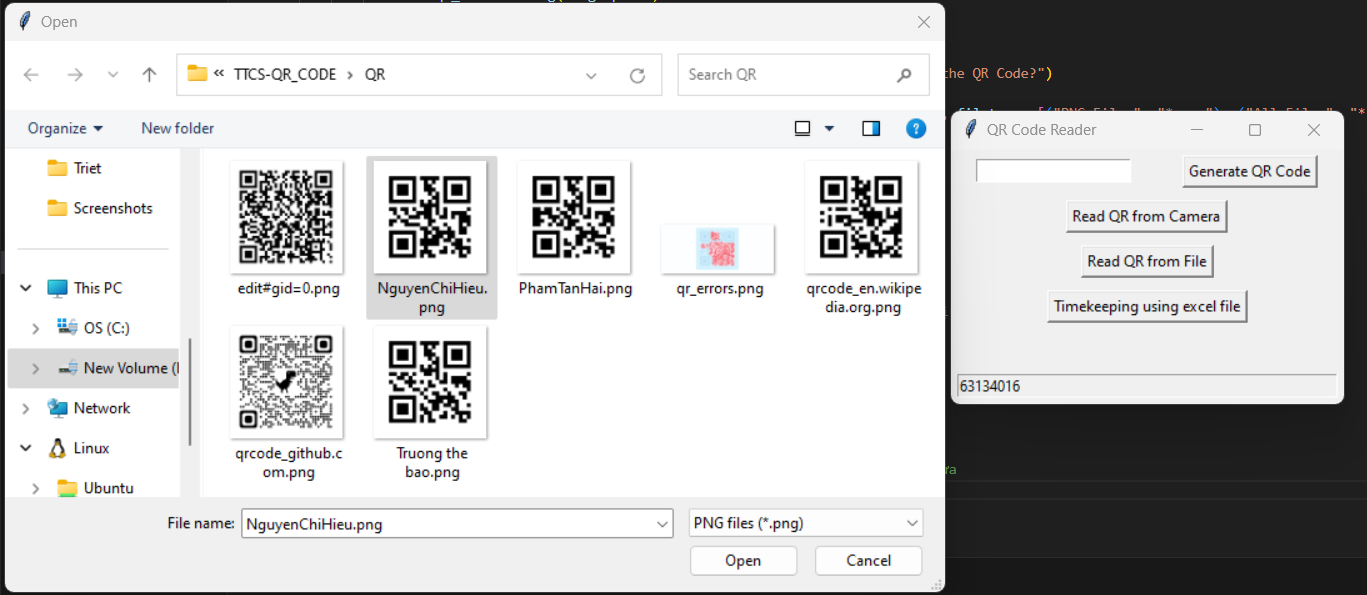
# Kết Quả



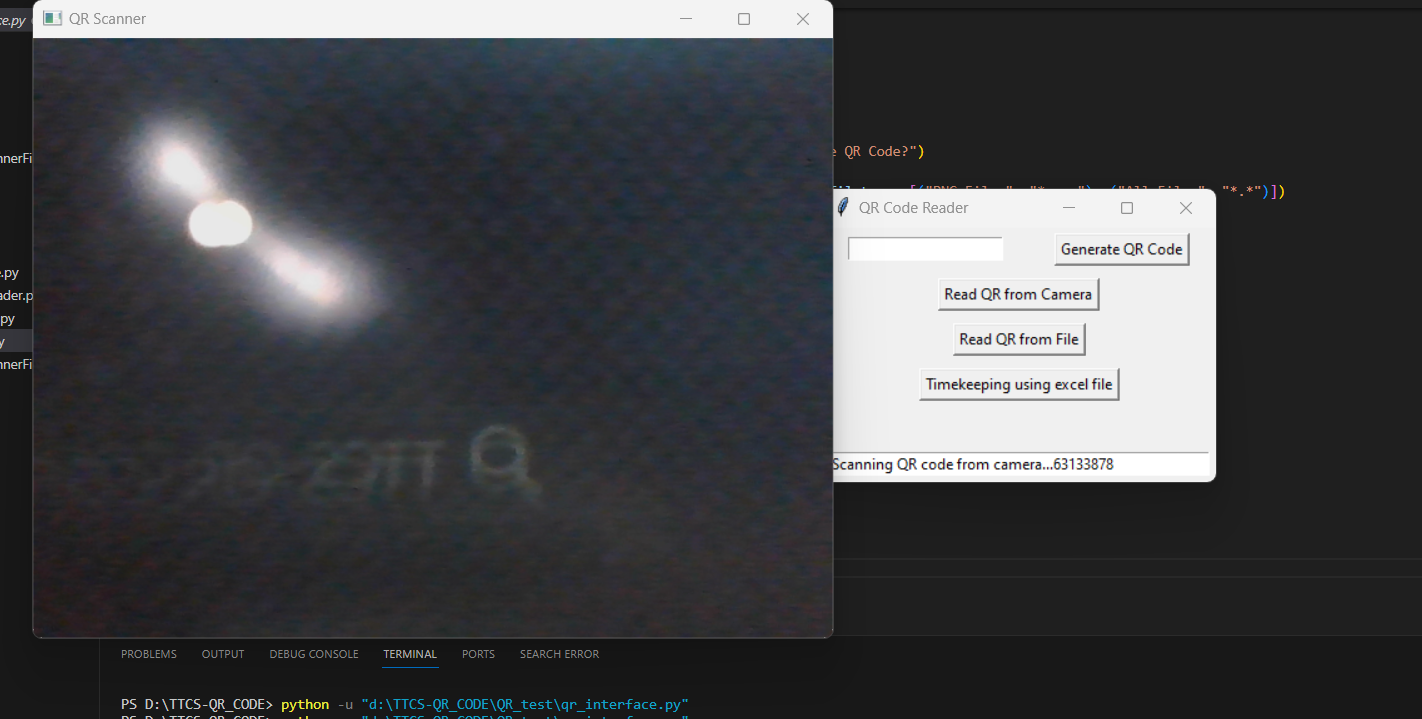
hình 3:1 Giao diện chính



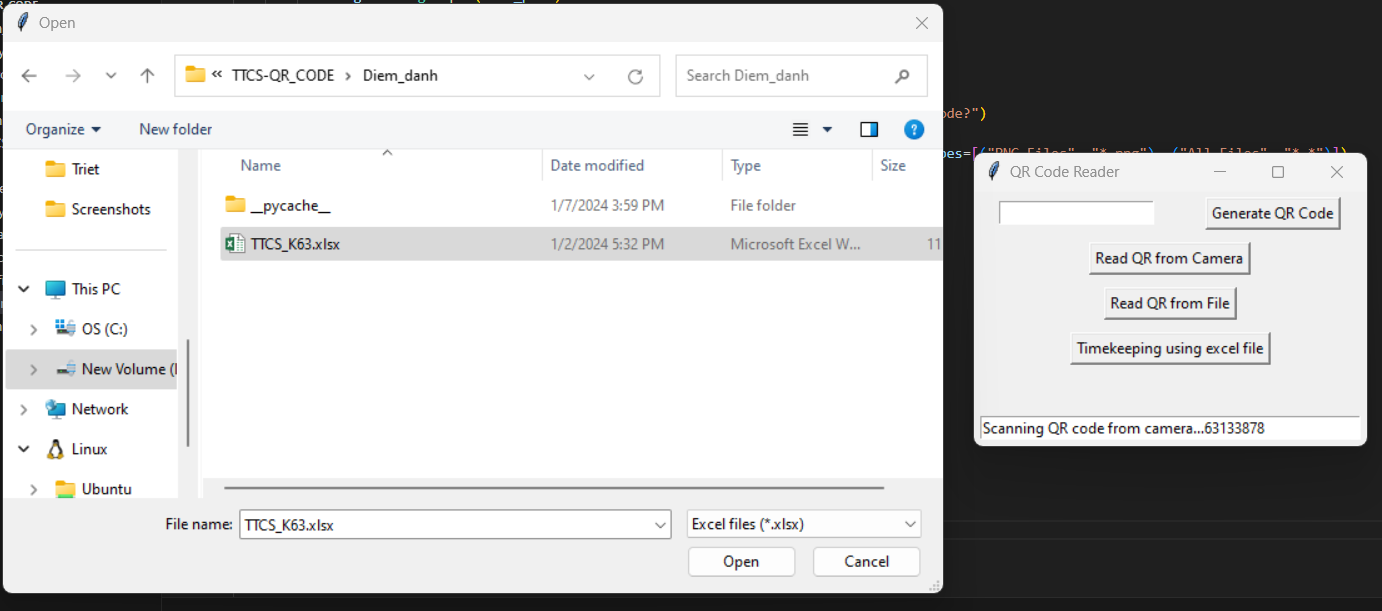
hình 3:2 Tạo mã QRCode



hình 3:3 Đọc mã QRCode từ file



hình 3:4 Đọc mã Qrcode từ camera



hình 3:5 Thực hiện chọn file điểm danh execl



hình 3:6 Mẫu định dạng file excel

# TÀI LIỆU THAM KHẢO

<https://www.geeksforgeeks.org/generate-qr-code-using-qrcode-in-python/>

<https://en.wikipedia.org/wiki/QR_code>

https://github.com/lincolnloop/python-qrcode