螢幕擷取grab\_screen([a,b,c,d])

輸入：

螢幕圖像

輸出：

特定範圍的螢幕圖像(BGRA)

參數：

([X軸初始座標,Y軸初始座標,X軸結束座標,Y軸結束座標])

生命偵測self\_blood\_count(a),boss\_blood\_count(b)

輸入：

圖像(灰度圖,二維陣列,uint8)

輸出：

整數(檢測到的生命值)

參數：

無

Reward判斷(包含死亡判斷)

action\_judge(boss\_blood, next\_boss\_blood, self\_blood, next\_self\_blood, stop, emergence\_break)

輸入：

boss\_blood(整數,敵人當前生命)

next\_boss\_blood(整數,敵人在我方做下個動作後的的生命)

self\_blood(整數,我方當前生命)

next\_self\_blood(整數,我方做下個動作後的的生命)

stop(uint8) (整數,避免連續讀取生命值，造成reward誤判的緩衝)

emergence\_break(整數,默認0,與緊急暫停訓練有關)

輸出：

reward(整數,動作回饋分數)

done(整數,與儲存資料到回放記憶體有關)

stop(整數,避免連續讀取生命值，造成reward誤判的緩衝)

emergence\_break(整數,與緊急暫停訓練有關)

參數：

無

DQN架構

agent = DQN(WIDTH, HEIGHT, action\_size, DQN\_model\_path, DQN\_log\_path)

輸入：

WIDTH(圖片寬,uint8)

HEIGHT(圖片高,uint8)

action\_size(動作數量 ,uint8)

DQN\_model\_path(訓練模型路徑, str)

DQN\_log\_path(訓練記錄檔案路徑, str)

輸出：

Agent(Class)

參數：

無

action = agent.Choose\_Action(station)

輸入：station陣列(88, 96, 1)

輸出：action 整數

參數：無

agent.save\_model()

輸入：無

輸出：無

參數：無

註：呼叫他能用來存檔

agent.Store\_Data(station, action, reward, next\_station, done)

輸入：

station 陣列(88, 96, 1)

action 整數

reward 整數

next\_station 陣列(88, 96, 1)

輸出：無

參數：done(uint8)

註：呼叫他能把當前狀態、下一個狀態跟reward儲存至回放記憶體

Train\_Network(big\_BATCH\_SIZE, num\_step)

輸入：

big\_BATCH\_SIZE(整數,默認64,與一次讀取回放記憶體的量有關)

num\_step(整數,計算做了幾個動作)

輸出：無

參數：無

agent.Update\_Target\_Network()

輸入：無

輸出：無

參數：無

註：將當前訓練的主模型網路複製到對抗模型網路裡

重新開始 restart()

輸入：無

輸出：無

參數：無

註：模仿鍵盤動作，將腳色復活，以進行下一輪訓練

偵測鍵盤輸入keys = key\_check()

輸入：鍵盤動作

輸出：keys(char)

參數：無

定義動作 take\_action(action)

輸入：action(整數)

輸出：模擬按鍵動作

參數：無