目錄

一、 簡介

1. **動機：**

一開始在訂定遊戲主題時，我們的想法就是以實現兒時回憶中的小遊戲為主，最初把目標設定為森林冰火人這款遊戲，是因為這款遊戲除了能夠考驗兩個人在遊玩遊戲時的溝通技巧及默契，遊戲中同時也存在很多機關，需要玩家自己去發現並且觸發它進而破關，在遊戲過程中充滿了新奇感，而且這款遊戲是我們童年相同的回憶。當然，小遊戲的類別也分為很多種，除了像森林冰火人雙人合作類型的小遊戲，也有雙人格鬥、單人闖關等眾多不同類型的小遊戲，期中過後，我們想了解格鬥遊戲的撰寫方式，而且相當憧憬火影忍者相關格鬥遊戲，因此我們把森林冰火人當中冰人與火人的角色，發揮創意想像成鳴人及佐助，為這次專題再訂下更多元的主題，設計出同時擁有雙人合作、雙人格鬥及單人闖關三種不同模式的遊戲。

2. 分工：

李承紘：Init、Over 程式撰寫、部分遊戲內程式撰寫、選單程式撰寫、素材整理及製作

歐陽文立：角色程式撰寫、遊戲主要部分撰寫、程式維護、素材尋找及製作

二、 遊戲介紹

1. 遊戲說明：

- 闖關模式(主選單:遊戲開始)：我們要幫助森林中的水女孩和火男孩順利過關並收集所有的寶石，而且每個關卡都會有讓玩家們動動腦的地方，必須靠兩個人的配合才能解決問題，不同顏色的陷阱只能容納不同顏色的人通過！(遊戲操作：以鍵盤 W、S、A、D 鍵控制火男孩移動，鍵盤方向鍵控制水女孩移動。)
- 雙人格鬥(主選單:對戰模式)：在風向及地形的限制之下，雙方玩家都不能掉進去毒坑，並且嘗試發動攻擊及技能，試圖把對方打進毒坑獲得勝利。(遊戲操作：以鍵盤 W、S、A、D 鍵控制火男孩移動、↓:防禦、1:攻擊、2:集氣；鍵盤方向鍵控制水女孩移動、S：防禦、J：攻擊、K：集氣。 奧義： 防禦+方向鍵+攻擊；突進：兩下方向鍵+攻擊。)
- 單人闖關(主選單:魔王模式)：共分為三個關卡，依序為鹿丸、寧次、大蛇丸，每一個關卡遇到的挑戰者都有不一樣的技能，玩家(佐助)要善用查克拉，並且適當的使用技能，打敗所有挑戰者，通過挑戰賽！(遊戲操作：X：普通攻擊、Z：跳躍、長按 X：蓄氣發

動技能、空白鍵：作弊恢復查克拉)

2. 遊戲圖形：

闖關模式冰人移動

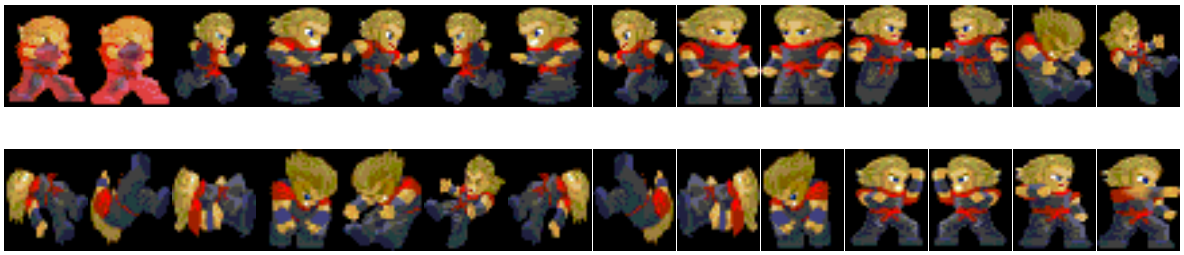
闖關模式火人移動

闖關模式時評分

闖關模式門

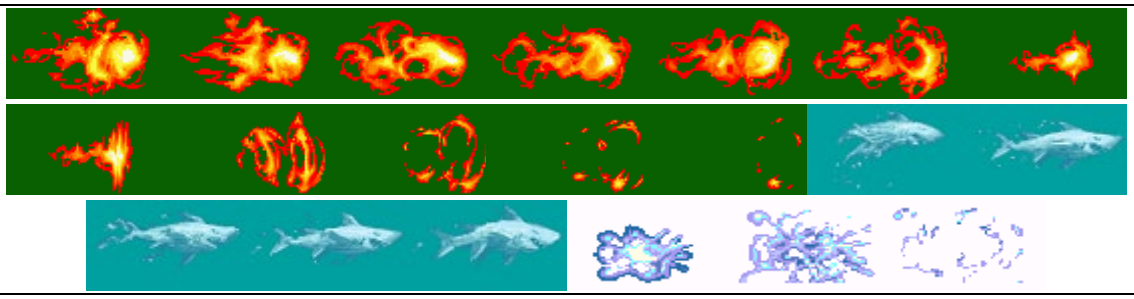
闖關模式物件

對戰模式火人

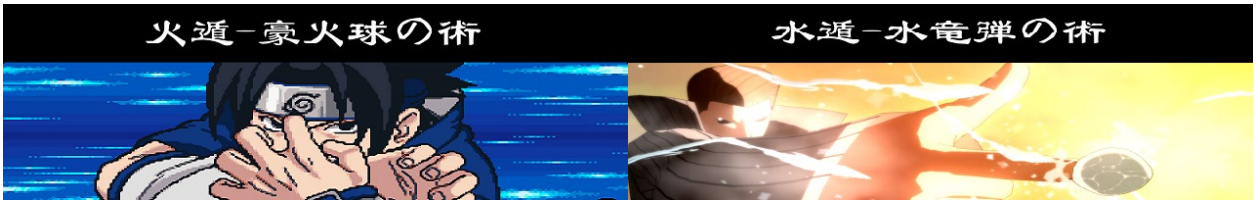


對戰模式冰人

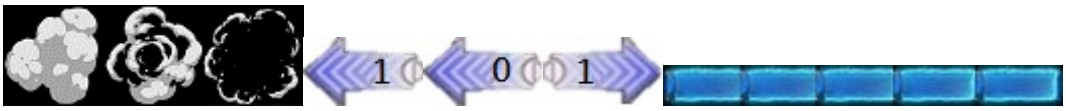
對戰模式技能



對戰模式技能動畫

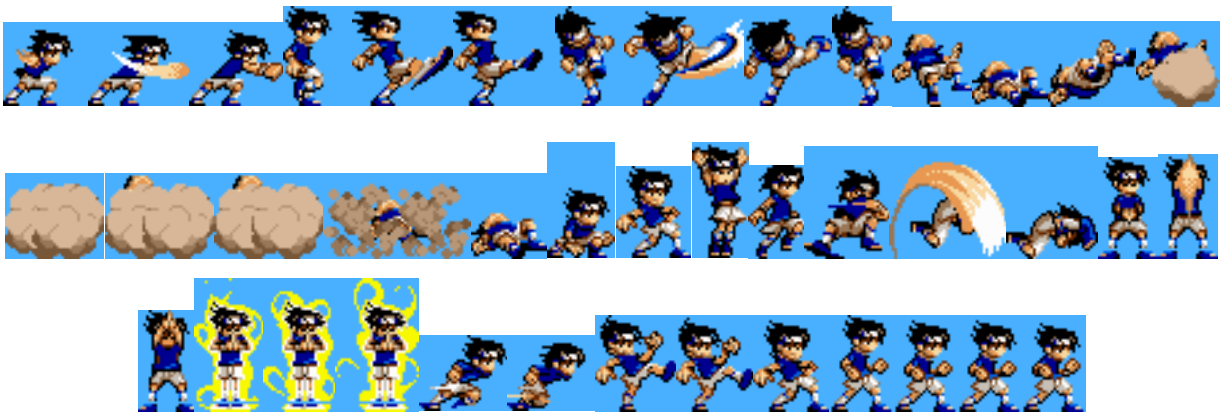


對戰開頭動畫、風向圖、UI

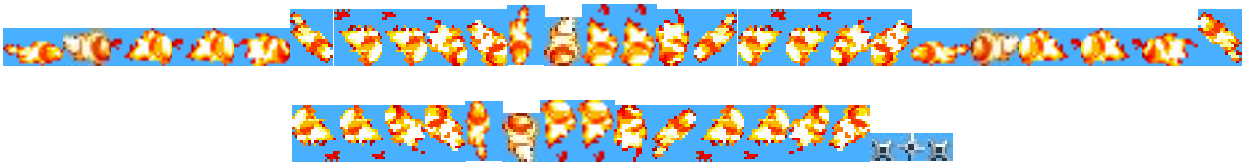


選關畫面評分

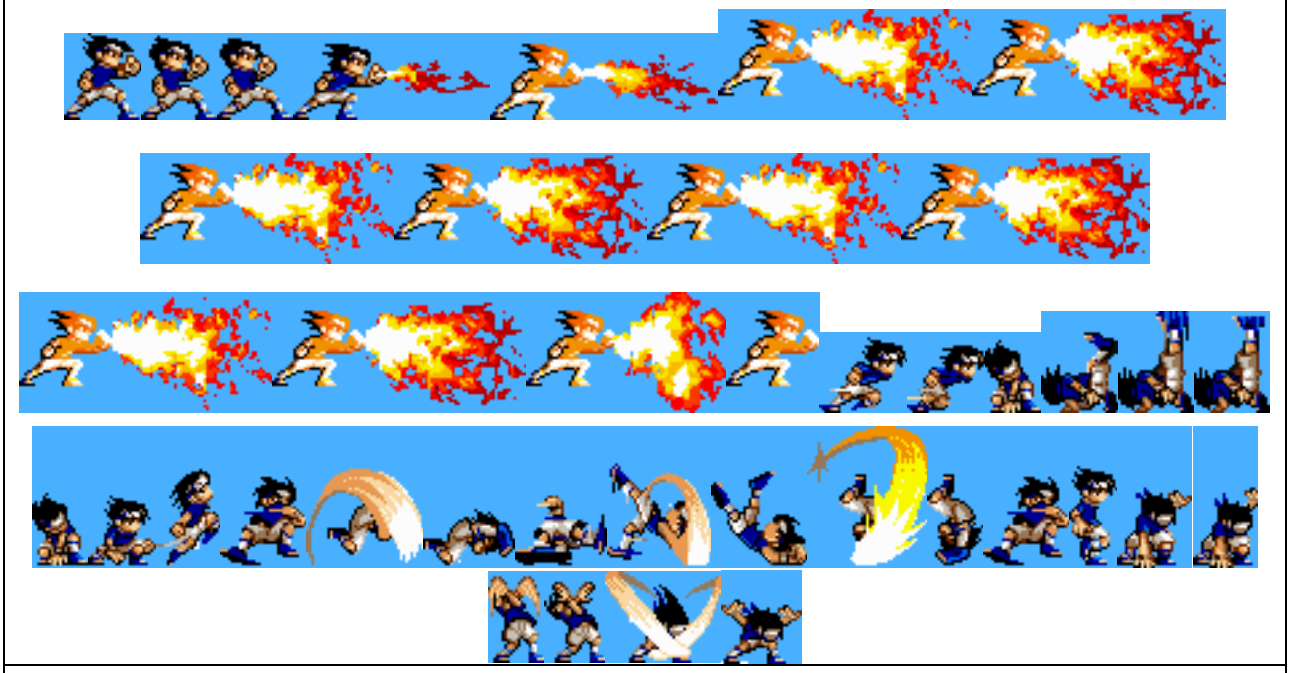
單人模式佐助



單人模式丟擲物

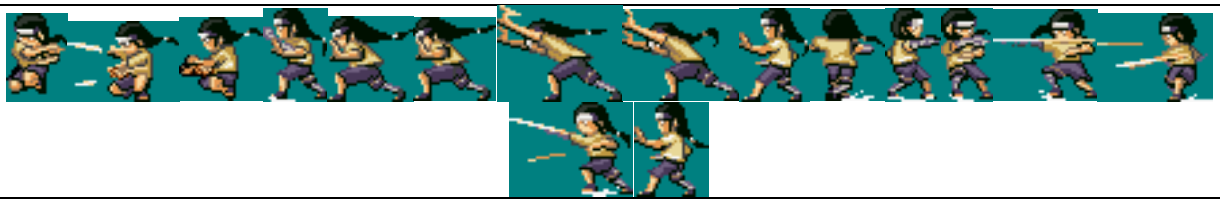


單人模式技能

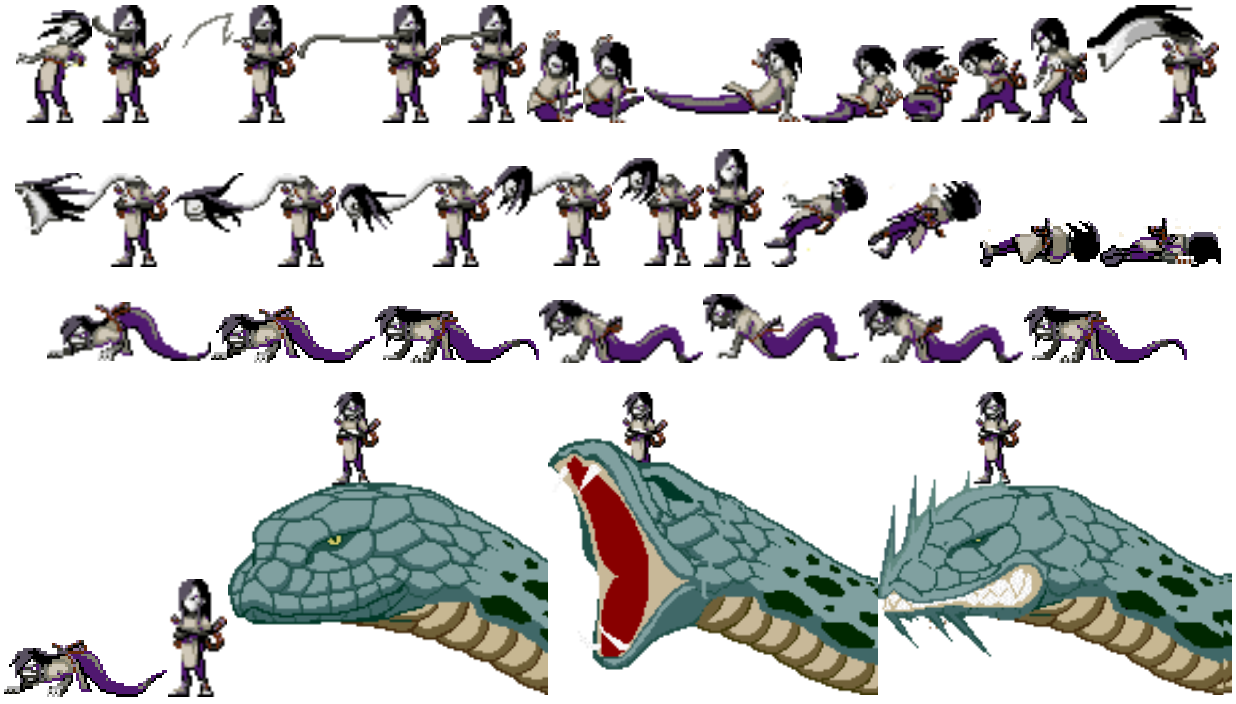


單人模式第一關魔王

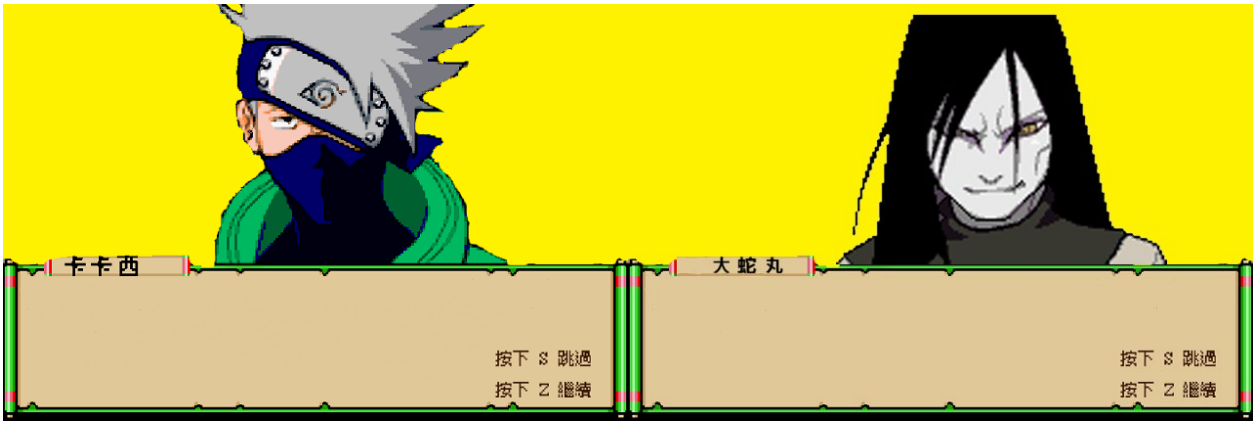
第二關魔王



第三關魔王



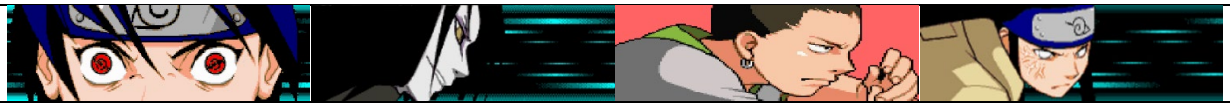
闖關模式開場



闖關模式 UI

多重手裏剣之術火遁　豪火球の術火遁　風仙花の术种子き え弾

技能橫幅



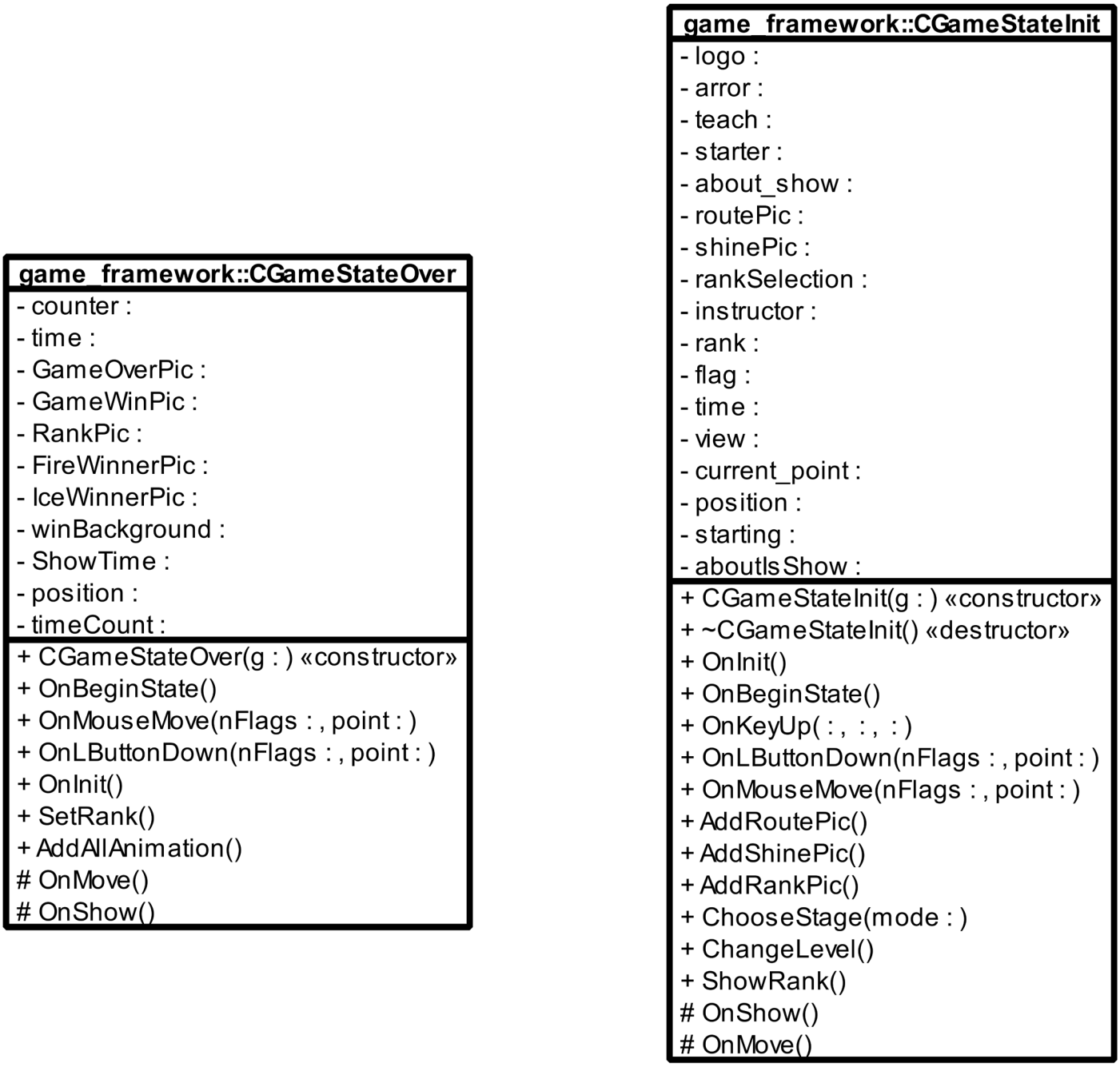
3. 遊戲音效：

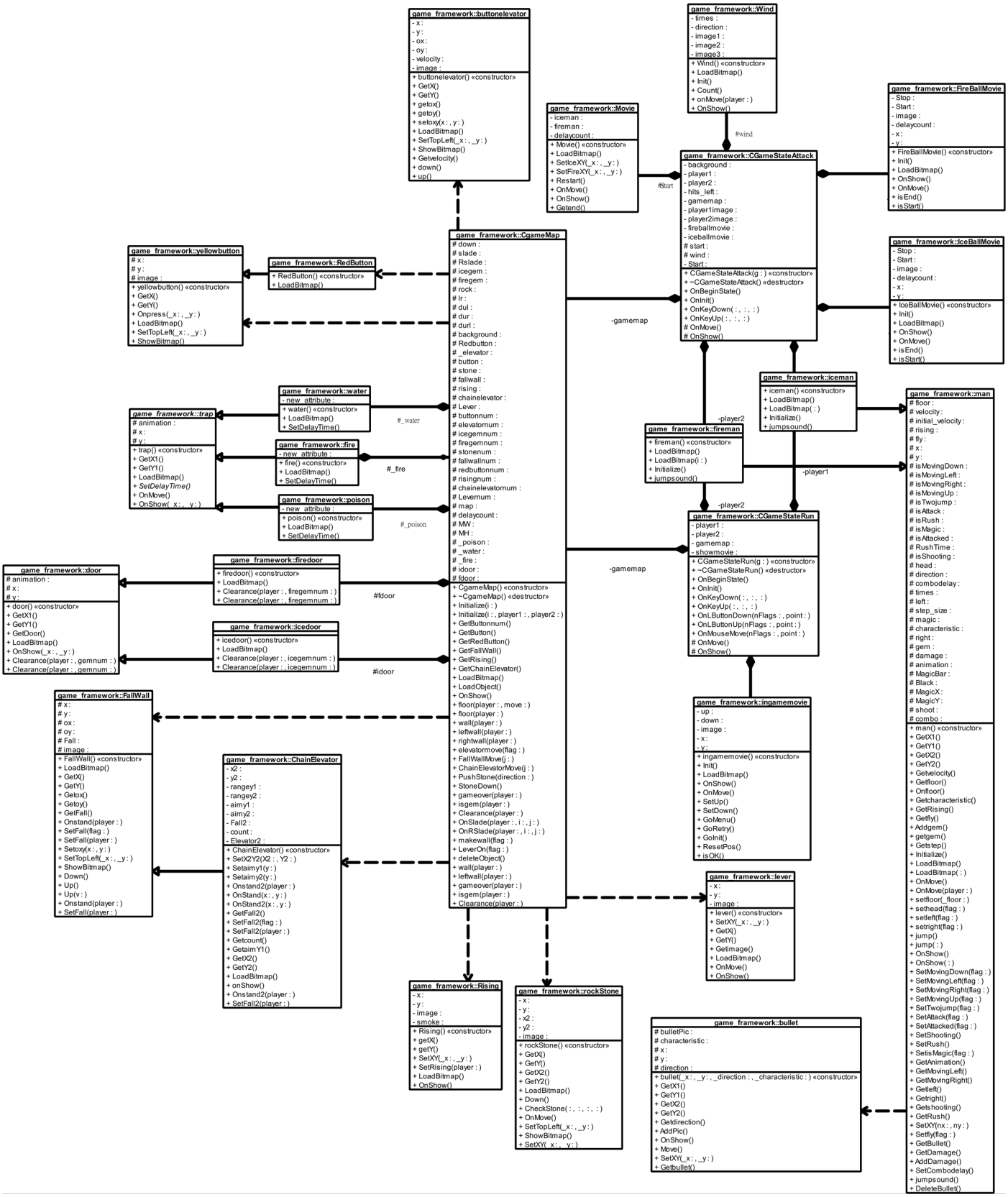
音樂名稱	使用物件	說明
attack	man	對戰模式中被擊中時音效
attackmusic	CGameStateAttack	對戰模式背景音樂
BGM	CGameStateNaruto	單人模式背景音樂
book	CGameStateInit	開頭解說畫面翻頁聲
CharacterDied	CGameStateAttack、CGameStateRun	闖關、對戰模式角色死亡
DiamondCollected	CGameStateRun	闖關模式吃到加分寶石聲
EnemySkill	CGameStateNaruto	技能前置動畫聲
fireball	Fireballmovie	對戰模式火球前置動畫聲
fireboy_jump	Fireman	火人跳躍聲
GameMusic	CGameStateRun	闖關模式背景音樂
Hover	CGameStateInit	開頭解說畫面按鈕聲
iceball	Iceballmovie	對戰模式冰球前置動畫聲
isAttacked	Sasuke、EnemyAI	單人模式被攻擊聲
isSkillsuccess	Sasuke	單人模式技能大成功聲
LevelFailed	CGameStateRun	闖關模式遊戲失敗音效
LevelSuccess	CGameStateRun	闖關模式遊戲成功音效
Magic	Sasuke、Man	對戰、單人模式集氣
Naruto2	CGameStateNaruto	大蛇丸 BGM
NarutoDie	Sasuke、EnemyAI	角色倒地音效
Round1	CGameStateAttack	開頭準備音效
SasukeAttack	Sasuke	角色揮拳聲
Sasukecombo1	Sasuke	角色吆喝聲
Sasukecombo2	Sasuke	角色吆喝聲 2
sasukefireball	Sasukefireball	單人模式火球聲
sikk4-1	Sasuke	佐助獅子連彈技能聲

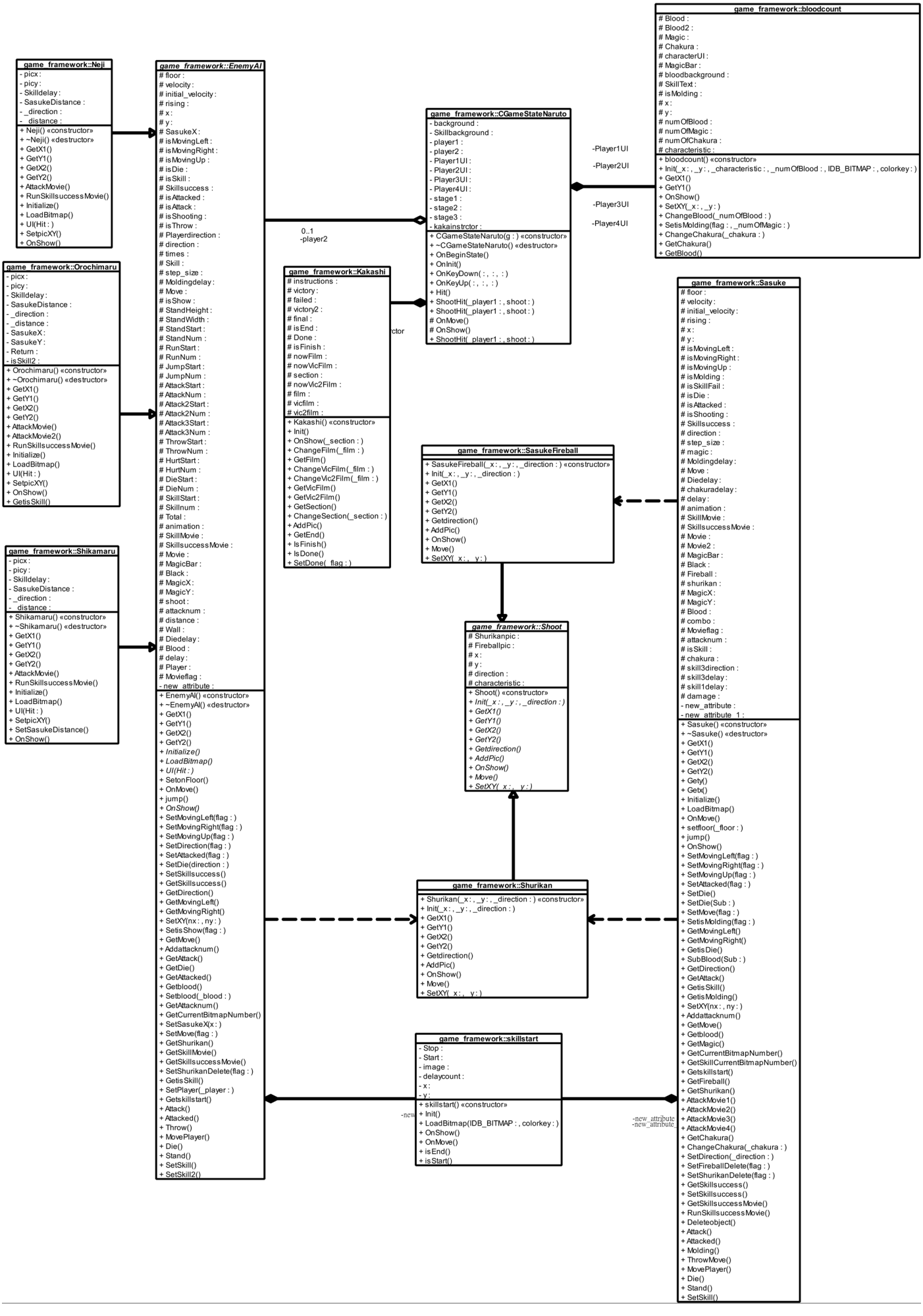
skill1	Sasuke	佐助技能成功聲
skill4-2	Sasuke	佐助獅子連彈攻擊聲
watergirl_jump	Iceman	冰人跳躍聲

三、 程式設計

1. 程式架構：




game_framework::buttonelevator
- x :
- y :
- ox :
- oy :
- velocity :
- image :
+ buttonelevator() «constructor»
+ GetX()
+ GetY()
+ getox()
+ getoy()
+ setoxy(x : , y :)
+ LoadBitmap()
+ SetTopLeft(_x : , _y :)
+ ShowBitmap()
+ Getvelocity()
+ down()
+ up()
game_framework::Wind
- times :
- direction :
- image1 :
- image2 :
- image3 :
+ Wind() «constructor»
+ LoadBitmap()
+ Init()
+ Count()
+ onMove(player :)
+ OnShow()
#wind
game_framework::Movie
- iceman :
- fireman :
- delaycount :
+ Movie() «constructor»
+ LoadBitmap()
+ SetIceXY(_x : , _y :)
+ SetFireXY(_x : , _y :)
+ Restart()
+ OnMove()
+ OnShow()
+ Getend()
#Start
game_framework::FireBallMovie
- Stop :
- Start :
- image :
- delaycount :
- x :
- y :
+ FireBallMovie() «constructor»
+ Init()
+ LoadBitmap()
+ OnShow()
+ OnMove()
+ isEnd()
+ isStart()
game_framework::CGameStateAttack
- background :
- player1 :
- player2 :
- hits_left :
- gamemap :
- player1image :
- player2image :
- fireballmovie :
- iceballmovie :
start :
wind :
- Start :
+ CGameStateAttack(g :) «constructor»
+ ~CGameStateAttack() «destructor»
+ OnBeginState()
+ OnInit()
+ OnKeyDown(: , : , :)
+ OnKeyUp(: , : , :)
# OnMove()
# OnShow()
game_framework::IceBallMovie
- Stop :
- Start :
- image :
- delaycount :
- x :
- y :
+ IceBallMovie() «constructor»
+ Init()
+ LoadBitmap()
+ OnShow()
+ OnMove()
+ isEnd()
+ isStart()
game_framework::yellowbutton
x :
y :
image :
+ yellowbutton() «constructor»
+ GetX()
+ GetY()
+ Onpress(_x : , _y :)
+ LoadBitmap()
+ SetTopLeft(_x : , _y :)
+ ShowBitmap()
game_framework::RedButton
+ RedButton() «constructor»
+ LoadBitmap()
game_framework::CgameMap
down :
slade :
Rslade :
icegem :
firegem :
rock :
lr :
dul :
dur :
durl :
background :
Redbutton :
_elevator :
button :
stone :
fallwall :
rising :
chainelevator :
Lever :
buttonnum :
elevatornum :
icegemnum :
firegemnum :
stonenum :
fallwallnum :
redbuttonnum :
risingnum :
chainelevatornum :
Levernum :
map :
delaycount :
MW :
MH :
_poison :
_water :
_fire :
idoor :
fdoor :
+ CgameMap() «constructor»
+ ~CgameMap() «destructor»
+ Initialize(i :)
+ Initialize(i : , player1 : , player2 :)
+ GetButtonnum()
+ GetButton()
+ GetRedButton()
+ GetFallWall()
+ GetRising()
+ GetChainElevator()
+ LoadBitmap()
+ LoadObject()
+ OnShow()
+ floor(player : , move :)
+ floor(player :)
+ wall(player :)
+ leftwall(player :)
+ rightwall(player :)
+ elevatormove(flag :)
+ FallWallMove(j :)
+ ChainElevatorMove(j :)
+ PushStone(direction :)
+ StoneDown()
+ gameover(player :)
+ isgem(player :)
+ Clearance(player :)
+ OnSlade(player : , i : , j :)
+ OnRSlade(player : , i : , j :)
+ makewall(flag :)
+ LeverOn(flag :)
+ deleteObject()
+ wall(player :)
+ leftwall(player :)
+ gameover(player :)
+ isgem(player :)
+ Clearance(player :)
-gamemap
-gamemap
game_framework::water
- new_attribute :
+ water() «constructor»
+ LoadBitmap()
+ SetDelayTime()
#_water
game_framework::trap
animation :
x :
y :
+ trap() «constructor»
+ GetX1()
+ GetY1()
+ LoadBitmap()
+ SetDelayTime()
+ OnMove()
+ OnShow(_x : , _y :)
game_framework::fire
- new_attribute :
+ fire() «constructor»
+ LoadBitmap()
+ SetDelayTime()
#_fire
game_framework::poison
- new_attribute :
+ poison() «constructor»
+ LoadBitmap()
+ SetDelayTime()
#_poison
game_framework::iceman
+ iceman() «constructor»
+ LoadBitmap()
+ LoadBitmap(:)
+ Initialize()
+ jumpsound()
-player2
game_framework::fireman
+ fireman() «constructor»
+ LoadBitmap()
+ LoadBitmap(i :)
+ Initialize()
+ jumpsound()
-player1
-player2
game_framework::man
floor :
velocity :
initial_velocity :
rising :
fly :
x :
y :
isMovingDown :
isMovingLeft :
isMovingRight :
isMovingUp :
isTwojump :
isAttack :
isRush :
isMagic :
isAttacked :
RushTime :
isShooting :
head :
direction :
combodelay :
times :
left :
step_size :
magic :
characteristic :
right :
gem :
damage :
animation :
MagicBar :
Black :
MagicX :
MagicY :
shoot :
combo :
+ man() «constructor»
+ GetX1()
+ GetY1()
+ GetX2()
+ GetY2()
+ Getvelocity()
+ Getfloor()
+ Onfloor()
+ Getcharacteristic()
+ GetRising()
+ Getfly()
+ Addgem()
+ getgem()
+ Getstep()
+ Initialize()
+ LoadBitmap()
+ LoadBitmap(:)
+ OnMove()
+ OnMove(player :)
+ setfloor(_floor :)
+ sethead(flag :)
+ setleft(flag :)
+ setright(flag :)
+ jump()
+ jump(:)
+ OnShow()
+ OnShow(:)
+ SetMovingDown(flag :)
+ SetMovingLeft(flag :)
+ SetMovingRight(flag :)
+ SetMovingUp(flag :)
+ SetTwojump(flag :)
+ SetAttack(flag :)
+ SetAttacked(flag :)
+ SetShooting()
+ SetRush()
+ SetisMagic(flag :)
+ GetAnimation()
+ GetMovingLeft()
+ GetMovingRight()
+ Getleft()
+ Getright()
+ Getshooting()
+ GetRush()
+ SetXY(nx : , ny :)
+ Setfly(flag :)
+ GetBullet()
+ GetDamage()
+ AddDamage()
+ SetCombodelay()
+ jumpsound()
+ DeleteBullet()
game_framework::CGameStateRun
- player1 :
- player2 :
- gamemap :
- showmovie :
+ CGameStateRun(g :) «constructor»
+ ~CGameStateRun() «destructor»
+ OnBeginState()
+ OnInit()
+ OnKeyDown(: , : , :)
+ OnKeyUp(: , : , :)
+ OnLButtonDown(nFlags : , point :)
+ OnLButtonUp(nFlags : , point :)
+ OnMouseMove(nFlags : , point :)
# OnMove()
# OnShow()
game_framework::door
animation :
x :
y :
+ door() «constructor»
+ GetX1()
+ GetY1()
+ GetDoor()
+ LoadBitmap()
+ OnShow(_x : , _y :)
+ Clearance(player : , gemnum :)
+ Clearance(player : , gemnum :)
game_framework::firedoor
+ firedoor() «constructor»
+ LoadBitmap()
+ Clearance(player : , firegemnum :)
+ Clearance(player : , firegemnum :)
#fdoor
game_framework::icedoor
+ icedoor() «constructor»
+ LoadBitmap()
+ Clearance(player : , icegemnum :)
+ Clearance(player : , icegemnum :)
#idoor
game_framework::FallWall
x :
y :
ox :
oy :
Fall :
image :
+ FallWall() «constructor»
+ LoadBitmap()
+ GetX()
+ GetY()
+ Getox()
+ Getoy()
+ GetFall()
+ Onstand(player :)
+ SetFall(flag :)
+ SetFall(player :)
+ Setoxy(x : , y :)
+ SetTopLeft(_x : , _y :)
+ ShowBitmap()
+ Down()
+ Up()
+ Up(v :)
+ Onstand(player :)
+ SetFall(player :)
game_framework::ChainElevator
- x2 :
- y2 :
- rangey1 :
- rangey2 :
- aimy1 :
- aimy2 :
- Fall2 :
- count :
- Elevator2 :
+ ChainElevator() «constructor»
+ SetX2Y2(X2 : , Y2 :)
+ Setaimy1(y :)
+ Setaimy2(y :)
+ Onstand2(player :)
+ OnStand(x : , y :)
+ OnStand2(x : , y :)
+ GetFall2()
+ SetFall2(flag :)
+ SetFall2(player :)
+ Getcount()
+ GetaimY1()
+ GetX2()
+ GetY2()
+ LoadBitmap()
+ onShow()
+ Onstand2(player :)
+ SetFall2(player :)
game_framework::ingamemovie
- up :
- down :
- image :
- x :
- y :
+ ingamemovie() «constructor»
+ Init()
+ LoadBitmap()
+ OnShow()
+ OnMove()
+ SetUp()
+ SetDown()
+ GoMenu()
+ GoRetry()
+ GoInit()
+ ResetPos()
+ isOK()
game_framework::lever
- x :
- y :
- image :
+ lever() «constructor»
+ SetXY(_x : , _y :)
+ GetX()
+ GetY()
+ Getimage()
+ LoadBitmap()
+ OnMove()
+ OnShow()
game_framework::Rising
- x :
- y :
- image :
- smoke :
+ Rising() «constructor»
+ getX()
+ getY()
+ SetXY(_x : , _y :)
+ SetRising(player :)
+ LoadBitmap()
+ OnShow()
game_framework::rockStone
- x :
- y :
- x2 :
- y2 :
- image :
+ rockStone() «constructor»
+ GetX()
+ GetY()
+ GetX2()
+ GetY2()
+ LoadBitmap()
+ Down()
+ CheckStone(: , : , : , :)
+ OnMove()
+ SetTopLeft(_x : , _y :)
+ ShowBitmap()
+ SetXY(x : , y :)
game_framework::bullet
bulletPic :
characteristic :
x :
y :
direction :
+ bullet(_x : , _y : , _direction : , _characteristic :) «constructor»
+ GetX1()
+ GetY1()
+ GetX2()
+ GetY2()
+ Getdirection()
+ AddPic()
+ OnShow()
+ Move()
+ SetXY(_x : , _y :)
+ Getbullet()


game_framework::Neji
- picx :
- picy :
- SkillDelay :
- SasukeDistance :
- _direction :
- _distance :
+ Neji() «constructor»
+ ~Neji() «destructor»
+ GetX1()
+ GetY1()
+ GetX2()
+ GetY2()
+ AttackMovie()
+ RunSkillsuccessMovie()
+ Initialize()
+ LoadBitmap()
+ UI(Hit :)
+ SetpicXY()
+ OnShow()
game_framework::EnemyAI
floor :
velocity :
initial_velocity :
rising :
x :
y :
SasukeX :
isMovingLeft :
isMovingRight :
isMovingUp :
isDie :
isSkill :
Skillsuccess :
isAttacked :
isAttack :
isShooting :
isThrow :
Playerdirection :
direction :
times :
Skill :
step_size :
Moldingdelay :
Move :
isShow :
StandHeight :
StandWidth :
StandStart :
StandNum :
RunStart :
RunNum :
JumpStart :
JumpNum :
AttackStart :
AttackNum :
Attack2Start :
Attack2Num :
Attack3Start :
Attack3Num :
ThrowStart :
ThrowNum :
HurtStart :
HurtNum :
DieStart :
DieNum :
SkillStart :
SkillNum :
Total :
animation :
SkillMovie :
SkillsuccessMovie :
Movie :
MagicBar :
Black :
MagicX :
MagicY :
shoot :
attacknum :
distance :
Wall :
Diedelay :
Blood :
delay :
Player :
Movieflag :
- new_attribute :
+ EnemyAI() «constructor»
+ ~EnemyAI() «destructor»
+ GetX1()
+ GetY1()
+ GetX2()
+ GetY2()
+ Initialize()
+ LoadBitmap()
+ UI(Hit :)
+ SetonFloor()
+ OnMove()
+ jump()
+ OnShow()
+ SetMovingLeft(flag :)
+ SetMovingRight(flag :)
+ SetMovingUp(flag :)
+ SetDirection(flag :)
+ SetAttacked(flag :)
+ SetDie(direction :)
+ SetSkillsuccess()
+ GetSkillsuccess()
+ GetDirection()
+ GetMovingLeft()
+ GetMovingRight()
+ SetXY(nx : , ny :)
+ SetisShow(flag :)
+ GetMove()
+ Addattacknum()
+ GetAttack()
+ GetDie()
+ GetAttacked()
+ Getblood()
+ Setblood(_blood :)
+ GetAttacknum()
+ GetCurrentBitmapNumber()
+ SetSasukeX(x :)
+ SetMove(flag :)
+ GetShurikan()
+ GetSkillMovie()
+ GetSkillsuccessMovie()
+ SetShurikanDelete(flag :)
+ GetisSkill()
+ SetPlayer(_player :)
+ Getskillstart()
+ Attack()
+ Attacked()
+ Throw()
+ MovePlayer()
+ Die()
+ Stand()
+ SetSkill()
+ SetSkill2()
game_framework::bloodcount
Blood :
Blood2 :
Magic :
Chakura :
characterUI :
MagicBar :
bloodbackground :
SkillText :
isMolding :
x :
y :
numOfBlood :
numOfMagic :
numOfChakura :
characteristic :
+ bloodcount() «constructor»
+ Init(_x : , _y : , _characteristic : , _numOfBlood : , IDB_BITMAP : , colorkey :)
+ GetX1()
+ GetY1()
+ OnShow()
+ SetXY(_x : , _y :)
+ ChangeBlood(_numOfBlood :)
+ SetisMolding(flag : , _numOfMagic :)
+ ChangeChakura(_chakura :)
+ GetChakura()
+ GetBlood()
-Player1UI
-Player2UI
-Player3UI
-Player4UI
game_framework::CGameStateNaruto
- background :
- Skillbackground :
- player1 :
- player2 :
- Player1UI :
- Player2UI :
- Player3UI :
- Player4UI :
- stage1 :
- stage2 :
- stage3 :
- kakainstrctor :
+ CGameStateNaruto(g :) «constructor»
+ ~CGameStateNaruto() «destructor»
+ OnBeginState()
+ OnInit()
+ OnKeyDown(: , : , :)
+ OnKeyUp(: , : , :)
+ Hit()
+ ShootHit(_player1 : , shoot :)
+ ShootHit(_player1 : , shoot :)
# OnMove()
# OnShow()
+ ShootHit(_player1 : , shoot :)
0..1
-player2
game_framework::Orochimaru
- picx :
- picy :
- SkillDelay :
- SasukeDistance :
- _direction :
- _distance :
- SasukeX :
- SasukeY :
- Return :
- isSkill2 :
+ Orochimaru() «constructor»
+ ~Orochimaru() «destructor»
+ GetX1()
+ GetY1()
+ GetX2()
+ GetY2()
+ AttackMovie()
+ AttackMovie2()
+ RunSkillsuccessMovie()
+ Initialize()
+ LoadBitmap()
+ UI(Hit :)
+ SetpicXY()
+ OnShow()
+ GetisSkill()
game_framework::Kakashi
instructions :
victory :
failed :
victory2 :
final :
isEnd :
Done :
isFinish :
nowFilm :
nowVicFilm :
section :
nowVic2Film :
film :
vicfilm :
vic2film :
+ Kakashi() «constructor»
+ Init()
+ OnShow(_section :)
+ ChangeFilm(_film :)
+ GetFilm()
+ ChangeVicFilm(_film :)
+ ChangeVic2Film(_film :)
+ GetVicFilm()
+ GetVic2Film()
+ GetSection()
+ ChangeSection(_section :)
+ AddPic()
+ GetEnd()
+ IsFinish()
+ IsDone()
+ SetDone(_flag :)
ctor
game_framework::Sasuke
floor :
velocity :
initial_velocity :
rising :
x :
y :
isMovingLeft :
isMovingRight :
isMovingUp :
isMolding :
isSkillFail :
isDie :
isAttacked :
isShooting :
Skillsuccess :
direction :
step_size :
magic :
Moldingdelay :
Move :
Diedelay :
chakuradelay :
delay :
animation :
SkillMovie :
SkillsuccessMovie :
Movie :
Movie2 :
MagicBar :
Black :
Fireball :
shurikan :
MagicX :
MagicY :
Blood :
combo :
Movieflag :
attacknum :
isSkill :
chakura :
skill3direction :
skill3delay :
skill1delay :
damage :
- new_attribute :
- new_attribute_1 :
+ Sasuke() «constructor»
+ ~Sasuke() «destructor»
+ GetX1()
+ GetY1()
+ GetX2()
+ GetY2()
+ Gety()
+ Getx()
+ Initialize()
+ LoadBitmap()
+ OnMove()
+ setfloor(_floor :)
+ jump()
+ OnShow()
+ SetMovingLeft(flag :)
+ SetMovingRight(flag :)
+ SetMovingUp(flag :)
+ SetAttacked(flag :)
+ SetDie()
+ SetDie(Sub :)
+ SetMove(flag :)
+ SetisMolding(flag :)
+ GetMovingLeft()
+ GetMovingRight()
+ GetisDie()
+ SubBlood(Sub :)
+ GetDirection()
+ GetAttack()
+ GetisSkill()
+ GetisMolding()
+ SetXY(nx : , ny :)
+ Addattacknum()
+ GetMove()
+ Getblood()
+ GetMagic()
+ GetCurrentBitmapNumber()
+ GetSkillCurrentBitmapNumber()
+ Getskillstart()
+ GetFireball()
+ GetShurikan()
+ AttackMovie1()
+ AttackMovie2()
+ AttackMovie3()
+ AttackMovie4()
+ GetChakura()
+ ChangeChakura(_chakura :)
+ SetDirection(_direction :)
+ SetFireballDelete(flag :)
+ SetShurikanDelete(flag :)
+ GetSkillsuccess()
+ SetSkillsuccess()
+ GetSkillsuccessMovie()
+ RunSkillsuccessMovie()
+ Deleteobject()
+ Attack()
+ Attacked()
+ Molding()
+ ThrowMove()
+ MovePlayer()
+ Die()
+ Stand()
+ SetSkill()
game_framework::SasukeFireball
+ SasukeFireball(_x : , _y : , _direction :) «constructor»
+ Init(_x : , _y : , _direction :)
+ GetX1()
+ GetY1()
+ GetX2()
+ GetY2()
+ Getdirection()
+ AddPic()
+ OnShow()
+ Move()
+ SetXY(_x : , _y :)
game_framework::Shoot
Shurikanpic :
Fireballpic :
x :
y :
direction :
characteristic :
+ Shoot() «constructor»
+ Init(_x : , _y : , _direction :)
+ GetX1()
+ GetY1()
+ GetX2()
+ GetY2()
+ Getdirection()
+ AddPic()
+ OnShow()
+ Move()
+ SetXY(_x : , _y :)
game_framework::Shurikan
+ Shurikan(_x : , _y : , _direction :) «constructor»
+ Init(_x : , _y : , _direction :)
+ GetX1()
+ GetY1()
+ GetX2()
+ GetY2()
+ Getdirection()
+ AddPic()
+ OnShow()
+ Move()
+ SetXY(_x : , _y :)
game_framework::Shikamaru
- picx :
- picy :
- SkillDelay :
- SasukeDistance :
- _direction :
- _distance :
+ Shikamaru() «constructor»
+ ~Shikamaru() «destructor»
+ GetX1()
+ GetY1()
+ GetX2()
+ GetY2()
+ AttackMovie()
+ RunSkillsuccessMovie()
+ Initialize()
+ LoadBitmap()
+ UI(Hit :)
+ SetpicXY()
+ SetSasukeDistance()
+ OnShow()
game_framework::skillstart
- Stop :
- Start :
- image :
- delaycount :
- x :
- y :
+ skillstart() «constructor»
+ Init()
+ LoadBitmap(IDB_BITMAP : , colorkey :)
+ OnShow()
+ OnMove()
+ isEnd()
+ isStart()
-new
-new_attribute
-new_attribute_1

2. 程式類別：

類別名稱	.h 檔行數	.cpp 檔行數	說明
bloodcount	31	140	單人模式血量 UI
bullet	26	142	對戰模式子彈
Cgamemap	72	1473	闖關模式及對戰模式地圖
Chainelevator	34	95	連鎖電梯物件
door	21	38	門物件繼承用
elevator	24	63	按鈕觸發電梯
EnemyAI	107	412	敵人 UI 基本操作(繼承用)
FallWall	33	83	石牆開關
Fire	15	27	火焰陷阱
Fireballmovie	19	58	對戰模式火球技能動畫
firedoor	15	47	火人通關的門
fireman	20	123	火人操作
Iceballmovie	18	55	對戰模式水龍技能動畫
icedoor	15	46	冰人通關的門
iceman	17	127	冰人操作
ingamemovie	25	89	闖關模式遊玩暫停動畫
Kakashi	37	314	單人模式開場對話
lever	21	43	闖關模式拉桿物件
man	97	646	闖關模式及對戰模式基本操作(繼承用)
mygame	195	1402	遊戲主體
Neji	28	696	單人模式第二關魔王 AI 操作及動畫
Orochimaru	35	640	單人模式第三關魔王 AI 操作及動畫
poison	15	27	毒池陷阱
RedButton	11	17	紅按鈕物件(控制石牆)
Rising	19	48	上升煙霧物件

Sasuke	113	1373	單人模式主角
SasukeFireball	23	134	單人模式火球物件(飛行道具)
Shikamaru	29	618	單人模式第一關魔王 AI 操作及動畫
Shoot	25	12	單人模式飛行道具繼承用
Shurikan	21	65	單人模式手里劍(飛行道具)
SkillStart	19	54	單人模式技能前置動畫
StartMovie	20	58	對戰模式開場變身動化
Stone	22	60	闖關模式石頭物件
trap	20	35	闖關模式陷阱(繼承用)
water	14	26	水池陷阱
Wind	19	51	對戰模式風向
yellowbutton	21	44	黃色按鈕(控制電梯)
總行數	1296	9381	

3. 程式技術：

A. 用中斷點及 TRACE 來進行偵錯及隨時掌握變數變化的過程來改進自身程式

B. 使用公式將兩張圖的誤差所計算出來，使素材大小不一也能不會出現偏移現象

C. 利用開檔讀檔的方式將地圖資訊從外部記事本當中以二維陣列的方式製作出遊戲地圖

D. 在所有量條方面，先顯示完整的量條，再根據其損失的比例，使用遮蔽圖來使得量條減少

E. 將敵人使用 EmenyAI 來進行主要功能的模板，再將其他敵人繼承這份模板，使得切換角色的部分可以使用指標的方式切換，大量減少了變數的宣告及程式行數的數量。

F. 時間計算方面使用 CSpecialEffect 的方式，將時間傳遞給其他的 state，也能擁有較為簡單的共同計時裝置。

G. 使用 CInteger 的方式將計時方面的事情以簡單的方式顯示出來，也使用了自己找尋到的素材來當作時間圖片。

H. 當需要大量相同圖片的時候，以同一個 CMovingBitmap 顯示在不同位置的方式，節省了記憶體也節省了變數宣告的麻煩

四、 結語

1. 問題及解決方法：

Q1：遇到 Memory leak。

A1：回朔到還沒有 Memory leak 的時候，去發現哪裡開始 Memory leak，找到後再去搜尋哪裡 Delete 失敗，找到後就很好進行 Delete。

Q2：素材大小不一，CAnimation 會偏移。

A2：CAnimation 的位置都以 Stand 為基準，利用公式將兩張圖的誤差所計算出來，在 OnShow 的時候將誤差也一併算入。

Q3：Class 互相 Include。

A3：遇見這問題是出現在單人模式中 Player1、Player2 兩方互相 Include，所以改成將雙方所需要的值在他們的上一層也就是 mygame 中傳入，來避免需要 Include 對方的情形。

Q4：CAnimation 的 Delay 調高會導致動作切換太慢，調低又會導致同一動作看不清分別。

A4：統一將 Delay 調為最低，再將需要看出分別的圖片多 Add 幾次，這樣便能自由地改變之間的 Delaytime 又不用多寫進裡面。

Q5：相似的 Class 但圖片不同。

A5：將相似的 Class 圖片分別用 virtual 進行引入，而 Load 進來的圖片只需告訴程式那些動作有幾張圖片便可自動融合進程式

Q6：碰撞條件相同的物件，但不同 Class。

A6：創造一個 Class 給雙方繼承，這樣便能在傳入 function 時將他們轉換成繼承後的 Class，便可進行判斷。

Q7：量條遞減動畫處理。

A7：量條採取公式並將其轉化成所需圖片數量，用圖片一格一格覆蓋過圖片，形成動畫遞減效果。

2. 時間表：

週次	歐陽文立(小時)	李承紘(小時)	說明
1	7	6	可移動角色、尋找素材、製作地圖背景
2	8	7	製作跳躍、平台、地圖、新增第二角色
3	9	6	製作陷阱以及加分寶石

4	20	16	增加障礙物，優化陷阱圖片，增加通關 UI
5	12	12	完成第二關地圖背景
6	11	10	關卡開頭結尾圖示，第二關地板墜落物件
7	10	9	增加第二關物件，完成第三關背景
8	11	9	完成第三關，完成關卡評價及通關紀錄
9	22	18	第四關地圖，關卡完成時間
10	22	19	對戰完成
11	12	12	第五關背景地圖及遊戲中重新開始按鍵
12	33	12	完成第五關 蹺蹺板物件
13	18	16	第一關敵人完成
14	56	31	單人模式完成
15	28	28	修正 BUG、撰寫報告
16	x	x	無
17	x	x	無
合計	279	211	

3. 貢獻比例：

李承紘：50%

歐陽文立：50%

4. 自我檢核表：

	項目	完成否	無法完成的原因
1	解決 Memory leak	☑已完成 ☐未完成	
2	自定遊戲 Icon	☑已完成 ☐未完成	
3	全螢幕啟動	☑已完成 ☐未完成	
4	有 About 畫面	☑已完成 ☐未完成	
5	初始畫面說明按鍵及滑鼠之用法與密技	☑已完成 ☐未完成	
6	上傳 setup/apk/source 檔	☑已完成 ☐未完成	

7	setup 檔可正確執行	☑已完成 ☐未完成	
8	報告字型、點數、對齊、行距、頁碼等格式正確	☑已完成 ☐未完成	
9	報告封面、側邊格式正確	☑已完成 ☐未完成	
10	報告附錄程式格式正確	☑已完成 ☐未完成	

5. 收穫：

李承紘：這學期在製作遊戲的過程中，我們使用了很多繼承與多型的技巧，像是 virtual function 的使用，把大二上學期老師所教導的技術實現，更熟悉物件導向程式設計的運用是我最大的收穫，而使用到最多的是遊戲中動畫技術-CAnimation，但為了使動畫呈現得更細緻素材整理也成為最耗時的環節，一個跳躍的畫面也許就需要 10 張的圖片需要處裡，很多 UI 介面我也使用 CAnimation 來做畫面的切換，使用起來更加快速，真正親自嘗試製作遊戲後，我也發現遊戲的音效是遊戲中很重要的元素，當角色在動作時配上音效，總是會有加分的效果。我們也利用開檔讀檔的技巧，把遊戲地圖中每一個物件的位置對應到數字，再把一連串的數字存在 txt 檔案中，如此一來程式匯入地圖的步驟就能更加簡潔。程式出錯的時候，sourcetree 的版本管控扮演了很重要的角色，在忘記每完成一次修改就編譯的情況下，要快速地使程式重新正常運作並 debug，就會使用 sourcetree 回復先前版本，在不確定程式執行的順序與效果時，利用中斷點可以確認當下每一個數值的狀態，而利用 TRACE 就能更動態的了解數值的改變情形。

歐陽文立：在這堂課程當中，我發現到了從頭建立起一項大工程的所需要花費的努力，在這門課以前，我常常在想，現在在學的是甚麼鬼東西，完全沒辦法想像運用到他的場面，或是想說這種寫法十分麻煩，非常難理解，但是經過這門課後，對於繼承或是類別的運用上，有著更深的了解了，從最開始的全部都寫在同一個 Class 當中，到後面慢慢地分割出一些 Class 來方便運用，到後面再進行化繁為簡到用一些單純可以給別人繼承用的 Class 來壓縮程式的複雜度及更方便的運用，感覺到能把二上所學習到的知識更加熟悉的發揮出來，也學習到了許多之前不會使用的方法，而程式撰寫方面，也感受到增加程式維護性的重要性，在剛開始寫的程式，程式維護度比較低，到後面修改或增加新功能的時候，都會感受到比較痛苦，到

後面剛開始準備寫程式的時候都會思考怎麼樣程式維護性會比較簡單或是比較容易進行更改，所以到後面在整理程式的時候就會感受到哪裡比較好修改，剛開始也常常遇到 Memory leak 的問題，但是越到後面搜尋速度也越來越快，所以感覺自己越來越會修改 Memory leak 的問題，而程式理解的部分，寫到後面程式理解度比較充足的時候，許多不必要的過程或是怎麼寫比較方便也會漸漸地增快撰寫速度，或是增快功能複雜度，運用這些，越到後期發現能寫的功能越來越多，發現前面能改良的寫法也越來越多，學習到了許多 Class 的運用，也感受到了繼承的方便之處，也更加精進了自己的程式能力，也學習到了運用中斷點，或是 TRACE 來對程式中怎麼運行來進行除錯，這堂課真的是收益良多。

6. 心得、感想：

李承紘：「物件導向程式設計實習」這門課程，讓我整個學期的生活變得更加豐富精采。每周六日坐在電腦前製作遊戲變成了一種習慣，也讓假日因為製作遊戲變得充實。從大二上學期第一次接觸 C++這個程式語言就覺得很有趣，把每一個 class 分割清楚，再藉由繼承、多型等技巧讓所有 class 的關係完整建立。我認為這種物件導向的概念使程式變得更簡單明瞭，也讓學習變得更加有趣。其中，製作遊戲可以完美的發揮物件導向的特性，遊戲中的角色、道具、機關都是物件，而一旦物件的數量越來越多，我們也必須更小心的定義物件之間的關係，包括每個物件之間的互動(回傳數值、偵測碰撞、更改 private 裡的變數等等)。在這一次的製作遊戲過程中，我也學到許多除了程式相關技能以外的能力。例如，面對製作遊戲過程產生的各種問題，我和我的組員常有不同的處理方式和思考模式，而我們會各自寫出自己的想法，甚至互相比較製作出來的成效，進而挑選出最合適的方法當作定案。在經過多次討論和溝通後，協調出的成果也通常比我們各自原來的想法好。透過這一次的實習課，我學到了團隊合作的力量、製作專案的相互磨合，以及許多溝通協調的重要性。同時，我也從我的組員身上學到許多，無論是判斷物件碰撞的方法或是實現遊戲物件效果的想法，感謝組員也謝謝這堂課讓我收穫不少。

歐陽文立：我覺得在這堂課當中，真的非常辛苦，算是我二下感受最難拿到的學分，但是感受卻十分的良好，因為感覺跟其他的課不同，這堂課可以感受到自己真的有很明顯的成長，在看著自己的程式慢慢地新增功能，慢慢地成為自己想像的樣子，慢慢地豐富程式的功能，讓自己感覺到自己的這份程式真的是獨一無二的，在每次 Demo 中拿著自己的程式，感受到

的不是驗收的壓力，而是想與他人分享自己程式功能的心情，每次做出新功能不是感覺到解決一項事情，而是感覺想要跟同學分享自己這項功能的開心，而在面對其他人也有能拿得出手的東西給別人看了，這份開心是只有認真地想要修完這堂課才感受的到的，而每天也是寫著這程式就不知不覺地到天亮，跟朋友一起寫這份程式，不但感受到一起成長的快樂，也會跟其他人比較自己那些功能比較酷炫，會互相分享自己的寫法，會互相分享自己發現的功能，這堂課修起來不誇張，真的是到現在最好玩的一堂課。

7. 對於本課程的建議：

附錄

mygame.h

```
#include"CgameMap.h"
#include"StartMovie.h"
#include"Wind.h"
#include "Fireballmovie.h"
#include"Iceballmovie.h"
#include"ingamemovie.h"
#include "Sasuke.h"
#include "Shikamaru.h"
#include "bloodcount.h"
#include "Neji.h"
#include "Shoot.h"
#include "Kakashi.h"
#include "Orochimaru.h"

enum AUDIO_ID                    // 定義各種音效的編號
{
    gamemusic=0,            // 0
    Gem,                    // 1
    GameOver,               // 2
    Icejump,                // 3
    Firejump,               // 4
    gamewin,                // 5
    hit,                    // 6
    fireball,               // 7
    iceball,                // 8
    attack,                 // 9
    attackmusic,            // 10
    Round1,                 // 1
    MagicFire,              // 2
    MagicIce,               // 3
    ButtonHover,            // 4
    Button2Hover,           // 5
    Button3Hover,           // 6
    Button4Hover,           // 7
    Button5Hover,           // 8
    Book,                   // 9
    NarutoBGM,              // 20
    SasukeCombo,            // 1
    SasukeCombo2,           // 2
    SasukeAttack,           // 3
    Sasukeskillsuccess,     // 4
    Sasukefireball,         // 5
    SasukeSkill,            // 6
    SasukeSkill4,           // 7
    SasukeSkill42,          // 8
    NarutoisAttacked,       // 9
    NarutoDie,              // 30
    EnemySkill,             // 31
    NarutoBGM2,             // 32
};
extern bool stage[5];           //關卡是否通關
extern int current_level;       //當前為第幾關
extern int ranklevel[5];
```

```
extern bool current_rank;
extern int winner;
namespace game_framework
{
	class CGameStateInit : public CGameState
	{
	public:
		CGameStateInit(CGame* g);
		~CGameStateInit();
		void OnInit();                                  // 遊戲的初值及圖形設定
		void OnBeginState();                            // 設定每次重玩所需的變數
		void OnKeyUp(UINT, UINT, UINT);                 // 處理鍵盤 Up 的動作
		void OnLButtonDown(UINT nFlags, CPoint point);  // 處理滑鼠的動作
		void OnMouseMove(UINT nFlags, CPoint point);
		void AddRoutePic();
		void AddShinePic();
		void AddRankPic();
		void ChooseStage(int mode);
		void ChangeLevel();
		void ShowRank();
	protected:
		void OnShow();                                  // 顯示這個狀態的遊戲畫面
		void OnMove();
	private:
		CMovingBitmap logo;                             // csie 的 logo
		CMovingBitmap arror;
		CMovingBitmap teach;
		CMovingBitmap starter;
		CMovingBitmap about_show;
		CAnimation routePic;
		CAnimation shinePic;
		CAnimation rankSelection;
		CAnimation instructor;
		vector<CAnimation*> rank;
		bool flag;
		int time;
		int view;
		int current_point;
		int position[5][2] = { {296, 436},{317, 381},{303, 319},{307, 255},{315,177} };
		bool starting;
		bool aboutIsShow;
	};
	class CGameStateRun : public CGameState
	{
	public:
		CGameStateRun(CGame* g);
		~CGameStateRun();
		void OnBeginState();                            // 設定每次重玩所需的變數
		void OnInit();                                  // 遊戲的初值及圖形設定
		void OnKeyDown(UINT, UINT, UINT);
		void OnKeyUp(UINT, UINT, UINT);
		void OnLButtonDown(UINT nFlags, CPoint point);  // 處理滑鼠的動作
		void OnLButtonUp(UINT nFlags, CPoint point);    // 處理滑鼠的動作
		void OnMouseMove(UINT nFlags, CPoint point);    // 處理滑鼠的動作

	protected:
		void OnMove();                                  // 移動遊戲元素
		void OnShow();                                  // 顯示這個狀態的遊戲畫面
	private:
		iceman		player1;
		fireman		player2;
		CgameMap	gamemap;
		ingamemovie	showmovie;
	};
	class CGameStateOver : public CGameState
	{
	public:
		CGameStateOver(CGame* g);
		void OnBeginState();                            // 設定每次重玩所需的變數
		void OnMouseMove(UINT nFlags, CPoint point);   // 處理滑鼠的動作
		void OnLButtonDown(UINT nFlags, CPoint point);
		void OnInit();
		void SetRank();
		void AddAllAnimation();
	protected:
		void OnMove();      // 移動遊戲元素
```

```
        void OnShow();
    private:
        int counter;    // 倒數之計數器
        int time;
        CAnimation GameOverPic;
        CAnimation GameWinPic;
        CAnimation RankPic;
        CAnimation FireWinnerPic;
        CAnimation IceWinnerPic;
        CMovingBitmap winBackground;
        vector<CAnimation*> ShowTime;
        int position[3][2] = { {300, 192},{345, 192},{390, 192}};
        CInteger timeCount;
    };
    class CGameStateAttack : public CGameState
    {
    public:
        CGameStateAttack(CGame* g);
        ~CGameStateAttack();
        void OnBeginState();                                    // 設定每次重玩所需的變數
        void OnInit();                                              // 遊戲的初值及圖形設定
        void OnKeyDown(UINT, UINT, UINT);
        void OnKeyUp(UINT, UINT, UINT);
    protected:
        void OnMove();                                                          // 移動遊戲元素
        void OnShow();                                  // 顯示這個狀態的遊戲畫面
    private:
        CMovingBitmap    background;                // 背景圖
        iceman      player1;                // 拍子
        fireman             player2;
        Movie               Start;
        Wind                wind;
        CInteger            hits_left;              // 剩下的撞擊數
        CgameMap            gamemap;
        CMovingBitmap    player1image;
        CMovingBitmap    player2image;
        FireBallMovie fireballmovie;
        IceBallMovie  iceballmovie;
    };
    class CGameStateNaruto : public CGameState
    {
    public:
        CGameStateNaruto(CGame* g);
        ~CGameStateNaruto();
        void OnBeginState();                                    // 設定每次重玩所需的變數
        void OnInit();                                              // 遊戲的初值及圖形設定
        void OnKeyDown(UINT, UINT, UINT);
        void OnKeyUp(UINT, UINT, UINT);
        bool Hit();
        bool ShootHit(Sasuke _player1, Shoot *shoot);
        bool ShootHit(EnemyAI* _player1, Shoot *shoot);
    protected:
        void OnMove();                                                          // 移動遊戲元素
        void OnShow();                                                          // 顯示這個狀態的遊戲畫面
    private:
        CMovingBitmap    background;                                    // 背景圖
        CAnimation          Skillbackground;
        Sasuke      player1;
        EnemyAI*            player2 = new Shikamaru;
        bloodcount          Player1UI,Player2UI, Player3UI,Player4UI;
        bool                stage1, stage2 ,stage3;
        Kakashi             kakainstrctor;
    };

}
```

mygame.cpp

```
bool stage[5] = { false,false,false,false,false };
int current_level = 1;
int ranklevel[5] = { 0,0,0,0,0 };
bool current_rank = false;
int winner ;
namespace game_framework
{

CGameStateInit::CGameStateInit(CGame* g)
```

```
    : CGameState(g)
{
}
CGameStateInit::~CGameStateInit()
{
	int rank_size = (int)rank.size();
	for (int i = 0; i < rank_size; i++)
	{
		delete rank[0];
		rank.erase(rank.begin());
	}
}
void CGameStateInit::AddShinePic()
{
	shinePic.AddBitmap(shining01, RGB(30, 30, 30));
	shinePic.AddBitmap(shining02, RGB(30, 30, 30));
	shinePic.AddBitmap(shining03, RGB(30, 30, 30));
	shinePic.AddBitmap(shining04, RGB(30, 30, 30));
	shinePic.AddBitmap(shining05, RGB(30, 30, 30));
	shinePic.AddBitmap(shining06, RGB(30, 30, 30));
	shinePic.AddBitmap(shining07, RGB(30, 30, 30));
	shinePic.AddBitmap(shining08, RGB(30, 30, 30));
	shinePic.AddBitmap(shining09, RGB(30, 30, 30));
	shinePic.AddBitmap(shining10, RGB(30, 30, 30));
	shinePic.AddBitmap(shining12, RGB(30, 30, 30));
	shinePic.AddBitmap(shining13, RGB(30, 30, 30));
	shinePic.AddBitmap(shining14, RGB(30, 30, 30));
	shinePic.AddBitmap(shining15, RGB(30, 30, 30));
	shinePic.AddBitmap(shining16, RGB(30, 30, 30));
	shinePic.AddBitmap(shining17, RGB(30, 30, 30));
	shinePic.AddBitmap(shining18, RGB(30, 30, 30));
	shinePic.AddBitmap(shining19, RGB(30, 30, 30));
	shinePic.AddBitmap(shining20, RGB(30, 30, 30));
}
void CGameStateInit::AddRankPic()
{
	for (unsigned int i = 0; i < rank.size(); i++)
	{
		rank[i]->AddBitmap(rank1, RGB(255,255,255));
		rank[i]->AddBitmap(rank2, RGB(255,255,255));
		rank[i]->AddBitmap(rank3, RGB(255,255,255));
		rank[i]->SetTopLeft(position[i][0], position[i][1]);
		rank[i]->SetDelayCount(1);
		rank[i]->Reset();
	}
}
void CGameStateInit::AddRoutePic()
{
	routePic.AddBitmap(route1);
	routePic.AddBitmap(routetomain);
	routePic.AddBitmap(route2);
	routePic.AddBitmap(route2tomain);
	routePic.AddBitmap(map3);
	routePic.AddBitmap(map3mainmenu);
	routePic.AddBitmap(map4);
	routePic.AddBitmap(map4mainmenu);
	routePic.AddBitmap(map5);
	routePic.AddBitmap(map5mainmenu);
	routePic.AddBitmap(map6);
	routePic.AddBitmap(map6mainmenu);
}
void CGameStateInit::OnInit()
{
	view = 0;
	current_point = 0;
	AddShinePic();
	AddRoutePic();
	teach.LoadBitmap(map3);
	teach.SetTopLeft(0, 0);
	logo.LoadBitmap(INIT);
	arror.LoadBitmap(ARROR,RGB(255,255,255));
	arror.SetTopLeft(175, 205);
	instructor.AddBitmap(INSTRUCT01);
	instructor.AddBitmap(INSTRUCT02);
	instructor.AddBitmap(INSTRUCT03);
	instructor.AddBitmap(INSTRUCT04);
```

```
	instructor.SetTopLeft(0, 0);
	instructor.SetDelayCount(1);
	instructor.Reset();
	routePic.SetTopLeft(0, 0);
	routePic.SetDelayCount(1);
	routePic.Reset();
	shinePic.SetTopLeft(283, 429);
	shinePic.SetDelayCount(1);
	shinePic.Reset();
	starter.LoadBitmap(starting_view);
	starter.SetTopLeft(100, 100);
	starting = true;
	about_show.LoadBitmap(about);
	about_show.SetTopLeft(0, 27);
	aboutIsShow = false;
	for (int temp = 0; temp < 5; temp++) rank.push_back(new CAnimation);
	AddRankPic();
	ChooseStage(1);
////////////////////////////////初始化音樂//////////////////////////////////////////////////
	CAudio::Instance()->Load(GameOver, "sounds\\LevelFailed.wav");
	CAudio::Instance()->Load(gamemusic, "sounds\\gamemusic.wav");
	CAudio::Instance()->Load(Gem, "sounds\\DiamondCollected.wav");
	CAudio::Instance()->Load(Icejump, "sounds\\watergirl_jump.wav");
	CAudio::Instance()->Load(Firejump, "sounds\\fireboy_jump.wav");
	CAudio::Instance()->Load(hit, "sounds\\hit.wav");
	CAudio::Instance()->Load(gamewin, "sounds\\LevelSuccess.wav");
	CAudio::Instance()->Load(fireball, "sounds\\fireball.mp3");
	CAudio::Instance()->Load(iceball, "sounds\\iceball.mp3");
	CAudio::Instance()->Load(attack, "sounds\\attack.mp3");
	CAudio::Instance()->Load(attackmusic, "sounds\\attackmusic.mp3");
	CAudio::Instance()->Load(Round1, "sounds\\Round1.mp3");
	CAudio::Instance()->Load(MagicFire, "sounds\\Magic.mp3");
	CAudio::Instance()->Load(MagicIce, "sounds\\Magic.mp3");
	CAudio::Instance()->Load(ButtonHover, "sounds\\Hover.mp3");
	CAudio::Instance()->Load(Button2Hover, "sounds\\Hover.mp3");
	CAudio::Instance()->Load(Button3Hover, "sounds\\Hover.mp3");
	CAudio::Instance()->Load(Button4Hover, "sounds\\Hover.mp3");
	CAudio::Instance()->Load(Button5Hover, "sounds\\Hover.mp3");
	CAudio::Instance()->Load(Book, "sounds\\book.wav");
	CAudio::Instance()->Load(NarutoBGM, "sounds\\BGM.wav");
	CAudio::Instance()->Load(SasukeCombo, "sounds\\SasukeCombo1.wav");
	CAudio::Instance()->Load(SasukeCombo2, "sounds\\SasukeCombo2.wav");
	CAudio::Instance()->Load(SasukeAttack, "sounds\\SasukeAttack.wav");
	CAudio::Instance()->Load(Sasukeskillsuccess, "sounds\\isSkillsuccess.wav");
	CAudio::Instance()->Load(Sasukefireball, "sounds\\sasukefireball.wav");
	CAudio::Instance()->Load(SasukeSkill, "sounds\\skill1.wav");
	CAudio::Instance()->Load(SasukeSkill4, "sounds\\sikk4-1.wav");
	CAudio::Instance()->Load(SasukeSkill42, "sounds\\skill4-2.wav");
	CAudio::Instance()->Load(NarutoisAttacked, "sounds\\isAttacked.wav");
	CAudio::Instance()->Load(NarutoDie, "sounds\\NarutoDie.wav");
	CAudio::Instance()->Load(EnemySkill, "sounds\\EnemySkill.wav");
	CAudio::Instance()->Load(NarutoBGM2, "sounds\\Naruto2.wav");
////////////////////////////////初始化音樂//////////////////////////////////////////////////
}
void CGameStateInit::OnBeginState()
{
	current_point = 0;
	flag = false;
	ChangeLevel();
	winner = 0;
}
void CGameStateInit::ChooseStage(int mode)
{
	if (stage[4] == true)
	{
		if (mode == 1) routePic.Movetonum(10);
		else routePic.Movetonum(11);
	}
	else if (stage[3] == true)
	{
		if (mode == 1) routePic.Movetonum(8);
		else routePic.Movetonum(9);
	}
	else if (stage[2] == true)
	{
		if (mode == 1) routePic.Movetonum(6);
```

```
            else routePic.Movetonum(7);
        }
        else if (stage[1] == true)
        {
            if (mode == 1) routePic.Movetonum(4);
            else routePic.Movetonum(5);
        }
        else if((stage[0] == true))
        {
            if (mode == 1) routePic.Movetonum(2);
            else routePic.Movetonum(3);
        }
        else
        {
            if (mode == 1) routePic.Movetonum(0);
            else routePic.Movetonum(1);
        }
        OnShow();
}
void CGameStateInit::ChangeLevel()
{
        for (unsigned int i = 0; i < rank.size(); i++) rank[i]->Movetonum(2-ranklevel[i]);
        current_rank = false;
}
void CGameStateInit::OnLButtonDown(UINT nFlags, CPoint point)
{
        if (starting == false && aboutIsShow == false)
        {
            if (view == 2)
            {
                if ((point.x > 388 && point.x < 456) && (point.y > 404 && point.y < 434))
                {
                    CAudio::Instance()->Play(Book, false);
                    if (instructor.IsFinalBitmap())
                    {
                        view = 0;
                        instructor.Reset();
                    }
                    else instructor.OnMove();
                }
            }
            else if (view == 0)
            {
                if (point.x >= 207 && point.y >= 201 && point.x <= 421 && point.y <= 241) view = 1;
                if (point.x >= 207 && point.y >= 259 && point.x <= 421 && point.y <= 298)
                {
                    current_level = 98;
                    GotoGameState(GAME_STATE_ATTACK);
                }
                if (point.x >= 207 && point.y >= 315 && point.x <= 421 && point.y <= 357) view = 2;
                if (point.x >= 207 && point.y >= 374 && point.x <= 421 && point.y <= 412)
                {
                    current_level = 98;
                    GotoGameState(GAME_STATE_NARUTO);
                }
                if (point.x >= 207 && point.y >= 429 && point.x <= 421 && point.y <= 464)
                {
                    PostMessage(AfxGetMainWnd()->m_hWnd, WM_CLOSE, 0, 0);
                }
                if (point.x >= 545 && point.y >= 0 && point.x <= 640 && point.y <= 90) aboutIsShow = true;
            }
            else
            {
                if ((point.x > 30 && point.x < 190) && (point.y > 425 && point.y < 450)) view = 0;
                else
                {
                    if ((view == 1) && ((point.x > 290 && point.x < 330) && (point.y > 430 && point.y < 470) ||
                    ((point.x > 310 && point.x < 350) && (point.y > 380 && point.y < 415)) ||
                    ((point.x > 300 && point.x < 335) && (point.y > 320 && point.y < 345)) ||
                    ((point.x > 302 && point.x < 335) && (point.y > 253 && point.y < 285)) ||
                    ((point.x > 314 && point.x < 347) && (point.y > 180 && point.y < 205))))
                    {
                        GotoGameState(GAME_STATE_RUN);
                    }
                }
            }
```

```
	}
}
void CGameStateInit::OnMouseMove(UINT nFlags, CPoint point)
{
	current_point = 0;
	if (starting == false && aboutIsShow == false)
	{

		if (view == 0)
		{
			if (point.x >= 207 && point.y >= 201 && point.x <= 421 && point.y <= 241)
			{
				if (arror.Top() != 205) CAudio::Instance()->Play(ButtonHover, false);
				arror.SetTopLeft(175, 205);
			}
			if (point.x >= 207 && point.y >= 259 && point.x <= 421 && point.y <= 298)
			{
				if (arror.Top() != 260) CAudio::Instance()->Play(Button2Hover, false);
				arror.SetTopLeft(175, 260);
			}
			if (point.x >= 207 && point.y >= 315 && point.x <= 421 && point.y <= 357)
			{
				if (arror.Top() != 320) CAudio::Instance()->Play(Button3Hover, false);
				arror.SetTopLeft(175, 320);
			}
			if (point.x >= 207 && point.y >= 374 && point.x <= 421 && point.y <= 412)
			{
				if (arror.Top() != 382) CAudio::Instance()->Play(Button4Hover, false);
				arror.SetTopLeft(175, 382);
			}
			if (point.x >= 207 && point.y >= 430 && point.x <= 421 && point.y <= 475)
			{
				if (arror.Top() != 434) CAudio::Instance()->Play(Button4Hover, false);
				arror.SetTopLeft(175, 434);
			}
			if (point.x >= 545 && point.y >= 0 && point.x <= 640 && point.y <= 90)
			{
				if (arror.Top() != 37) CAudio::Instance()->Play(Button4Hover, false);
				arror.SetTopLeft(528, 37);

			}
		}
		else if (view == 1)
		{
			if ((point.x > 30 && point.x < 190) && (point.y > 425 && point.y < 450)) ChooseStage(0);
			else
			{
				if (((point.x > 290 && point.x < 330) && (point.y > 430 && point.y < 470)))
				{
					current_point = 1;
					shinePic.SetTopLeft(283, 429);
					current_level = 1;
				}
				else if ((stage[0] == true) && ((point.x > 310 && point.x < 350) && (point.y > 380 && point.y < 415)))
				{
					current_point = 1;
					current_level = 2;
					shinePic.SetTopLeft(305, 372);
				}
				else if ((stage[1] == true) && ((point.x > 300 && point.x < 335) && (point.y > 320 && point.y < 345)))
				{
					current_point = 1;
					shinePic.SetTopLeft(293, 310);
					current_level = 3;
				}
				else if ((stage[2] == true) && ((point.x > 302 && point.x < 335) && (point.y > 253 && point.y < 285)))
				{
					current_point = 1;
					shinePic.SetTopLeft(294, 244);
					current_level = 4;
				}
				else if ((stage[3] == true) && ((point.x > 314 && point.x < 347) && (point.y > 180 && point.y < 205)))
				{
					current_point = 1;
					shinePic.SetTopLeft(305, 168);
					current_level = 5;
```

```
                                }
                                ChooseStage(1);
                        }
                }
                else if (view == 2)
                {
                        if ((point.x > 388 && point.x < 456) && (point.y > 404 && point.y < 434))
                        {
                                if (!flag) CAudio::Instance()->Play(ButtonHover + instructor.GetCurrentBitmapNumber(), false);
                                flag = true;
                        }
                        else flag = false;
                }
        }

}
void CGameStateInit::OnKeyUp(UINT nChar, UINT nRepCnt, UINT nFlags)
{
        const char KEY_Q = 0x51;
        const char KEY_Z = 0x5a;
        const char KEY_ENTER = 0x0D;
        if (nChar == KEY_Q) for (int i = 0; i < 5; i++)    stage[i] = true;
        else if (nChar == KEY_Z)
        {
                current_level = 99;
                GotoGameState(GAME_STATE_RUN);
        }
        else if (nChar == KEY_ENTER)
        {
                starting = false;
                aboutIsShow = false;
        }
}

void CGameStateInit::ShowRank()
{
        for (unsigned int i = 0; i < rank.size(); i++) if(stage[i] == true) rank[i]->OnShow();
}

void CGameStateInit::OnShow()
{
        if (view == 0)
        {
                logo.SetTopLeft(0, 0);
                logo.ShowBitmap();
                arror.ShowBitmap();
        }
        else if (view == 1)
        {
                routePic.OnShow();
                ShowRank();
                if (current_point == 1) shinePic.OnShow();
        }
        else if (view == 2) instructor.OnShow();

        if (starting) {
                starter.ShowBitmap();
        }
        if (aboutIsShow)
        {
                about_show.ShowBitmap();
        }
}

void CGameStateInit::OnMove()
{
        if (current_point == 1)
        {
                shinePic.OnMove();
                OnShow();
        }
}

///////////////////////////////////////////////////////////////////////////
// 這個 class 為遊戲的結束狀態(Game Over)
///////////////////////////////////////////////////////////////////////////
```

```
CGameStateOver::CGameStateOver(CGame* g)
	: CGameState(g)
{
}

void CGameStateOver::OnMove()
{
	if (winner == 2)
	{
		IceWinnerPic.OnMove();
		if (IceWinnerPic.IsFinalBitmap())GotoGameState(GAME_STATE_INIT);
	}
	else if (winner == 1)
	{
		FireWinnerPic.OnMove();
		if (FireWinnerPic.IsFinalBitmap())GotoGameState(GAME_STATE_INIT);
	}
	OnShow();
}
void CGameStateOver::SetRank()
{
	if (time < 60)
	{
		RankPic.Movetonum(0);
		ranklevel[current_level - 1] = 0;
	}
	else if (time >= 60 && time < 80)
	{
		RankPic.Movetonum(1);
		ranklevel[current_level - 1] = 1;
	}
	else
	{
		RankPic.Movetonum(2);
		ranklevel[current_level - 1] = 2;
	}
}
void CGameStateOver::OnBeginState()
{
	time = CSpecialEffect::GetCurrentTimeCount() / 30;
	counter = 30000 * 5;
	timeCount.SetInteger(time);
	IceWinnerPic.Reset();
	FireWinnerPic.Reset();
	SetRank();
}
void CGameStateOver::OnMouseMove(UINT nFlags, CPoint point)
{
	if (current_rank == false)
	{
		if ((point.x > 184 && point.x < 481) && (point.y > 219 && point.y < 251))		GameOverPic.Movetonum(1);
		else if ((point.x > 239 && point.x < 411) && (point.y > 300 && point.y < 375))	GameOverPic.Movetonum(2);
		else GameOverPic.Movetonum(0);
	}
	else
	{
		if ((point.x > 261 && point.x < 390) && (point.y > 325 && point.y < 349))		GameWinPic.Movetonum(1);
		else GameWinPic.Movetonum(0);
	}
	OnShow();
}
void CGameStateOver::OnLButtonDown(UINT nFlags, CPoint point)
{
	if (current_rank == false && winner == 0)
	{
		if ((point.x > 184 && point.x < 481) && (point.y > 219 && point.y < 251))
		{
			if(current_level != 98) GotoGameState(GAME_STATE_RUN);
			else GotoGameState(GAME_STATE_ATTACK);
		}
		else if ((point.x > 239 && point.x < 411) && (point.y > 300 && point.y < 375)) GotoGameState(GAME_STATE_INIT);
		else GameOverPic.Movetonum(0);
	}
	else if (current_rank == true && winner == 0)
	{
```

```
            if ((point.x > 261 && point.x < 390) && (point.y > 325 && point.y < 349)) GotoGameState(GAME_STATE_INIT);
        }

}

void CGameStateOver::AddAllAnimation()
{
        IceWinnerPic.AddBitmap(ICEATTACKED1, RGB(0, 0, 0));                    //16
        IceWinnerPic.AddBitmap(ICEATTACKED2, RGB(0, 0, 0));                    //17
        IceWinnerPic.AddBitmap(ICEATTACKED3, RGB(0, 0, 0));                    //18
        IceWinnerPic.AddBitmap(ICEATTACKED4, RGB(0, 0, 0));                    //19
        IceWinnerPic.AddBitmap(ICEATTACKED5, RGB(0, 0, 0));                    //20
        IceWinnerPic.AddBitmap(ICEATTACKED6, RGB(0, 0, 0));                    //21
        IceWinnerPic.AddBitmap(ICEATTACKED1, RGB(0, 0, 0));                    //16
        IceWinnerPic.AddBitmap(ICEATTACKED2, RGB(0, 0, 0));                    //17
        IceWinnerPic.AddBitmap(ICEATTACKED3, RGB(0, 0, 0));                    //18
        IceWinnerPic.AddBitmap(ICEATTACKED4, RGB(0, 0, 0));                    //19
        IceWinnerPic.AddBitmap(ICEATTACKED5, RGB(0, 0, 0));                    //20
        IceWinnerPic.AddBitmap(ICEATTACKED6, RGB(0, 0, 0));                    //21
        FireWinnerPic.AddBitmap(FIRE_ATTTACKED1, RGB(0, 0, 0));        //16
        FireWinnerPic.AddBitmap(FIRE_ATTTACKED2, RGB(0, 0, 0));        //17
        FireWinnerPic.AddBitmap(FIRE_ATTTACKED3, RGB(0, 0, 0));        //18
        FireWinnerPic.AddBitmap(FIRE_ATTTACKED4, RGB(0, 0, 0));        //19
        FireWinnerPic.AddBitmap(FIRE_ATTTACKED5, RGB(0, 0, 0));        //20
        FireWinnerPic.AddBitmap(FIRE_ATTTACKED6, RGB(0, 0, 0));        //21
        FireWinnerPic.AddBitmap(FIRE_ATTTACKED1, RGB(0, 0, 0));        //16
        FireWinnerPic.AddBitmap(FIRE_ATTTACKED2, RGB(0, 0, 0));        //17
        FireWinnerPic.AddBitmap(FIRE_ATTTACKED3, RGB(0, 0, 0));        //18
        FireWinnerPic.AddBitmap(FIRE_ATTTACKED4, RGB(0, 0, 0));        //19
        FireWinnerPic.AddBitmap(FIRE_ATTTACKED5, RGB(0, 0, 0));        //20
        FireWinnerPic.AddBitmap(FIRE_ATTTACKED6, RGB(0, 0, 0));        //21
        FireWinnerPic.SetTopLeft(307, 210);
        FireWinnerPic.SetDelayCount(8);
        IceWinnerPic.SetTopLeft(307, 210);
        IceWinnerPic.SetDelayCount(8);
}

void CGameStateOver::OnInit()
{
    ShowInitProgress(66);   // 接個前一個狀態的進度，此處進度視為 66%
        AddAllAnimation();
        winBackground.LoadBitmap(winnerBack);
        winBackground.SetTopLeft(0, 0);
        GameOverPic.AddBitmap(GameOver01);
        GameOverPic.AddBitmap(GameOver02);
        GameOverPic.AddBitmap(GameOver03);
        GameOverPic.SetDelayCount(1);
        GameOverPic.Reset();
        GameOverPic.SetTopLeft(0, 0);
        GameWinPic.AddBitmap(score);
        GameWinPic.AddBitmap(score2);
        GameWinPic.SetDelayCount(1);
        GameWinPic.Reset();
        GameWinPic.SetTopLeft(0, 0);
        RankPic.AddBitmap(CH_A, RGB(255, 255, 255));
        RankPic.AddBitmap(CH_B, RGB(255, 255, 255));
        RankPic.AddBitmap(CH_C, RGB(255, 255, 255));
        RankPic.SetDelayCount(1);
        RankPic.Reset();
        RankPic.SetTopLeft(347,252);
        timeCount.LoadBitmap();
        timeCount.SetTopLeft(280, 172);
    ShowInitProgress(100);
}
void CGameStateOver::OnShow()
{
        if (winner == 0)
        {
                if (!current_rank)
                {
                        GameOverPic.SetTopLeft(0, 0);
                        GameOverPic.OnShow();
                }
                else if (current_rank)
                {
                        GameWinPic.SetTopLeft(0, 0);
```

```
                GameWinPic.OnShow();
                timeCount.ShowBitmap();
                RankPic.OnShow();
            }
        }
        else if (winner == 2)
        {
            winBackground.ShowBitmap();
            IceWinnerPic.OnShow(2);
        }
        else if (winner == 1)
        {
            winBackground.ShowBitmap();
            FireWinnerPic.OnShow(2);
        }
}
CGameStateRun::CGameStateRun(CGame* g)
    : CGameState(g)
{
}
CGameStateRun::~CGameStateRun()
{
}
void CGameStateRun::OnBeginState()
{
    player1.Initialize();
    player2.Initialize();
    CSpecialEffect::GetCurrentTimeCount();
    gamemap.deleteObject();
    gamemap.Initialize(current_level,&player1,&player2);
    gamemap.LoadObject();
    CAudio::Instance()->Play(gamemusic);
}
void CGameStateRun::OnMove()
{
    showmovie.OnMove();
    if (showmovie.isOK())
    {
        ////////////////初始化變數/////////////////
        bool winfire, winice;
        bool press = false;
        bool Redpress = false;
        player1.Setfly(false);
        player2.Setfly(false);
        ////////////////////////////////////////////判斷是否在物件上/////////////////////////////////////
        for (int i = 0; i < (int)gamemap.GetFallWall().size(); i++) gamemap.GetFallWall()[i]->SetFall(0);
        for (int i = 0; i < (int)gamemap.GetChainElevator().size(); i++)
        {
            gamemap.GetChainElevator()[i]->SetFall(0);
            gamemap.GetChainElevator()[i]->SetFall2(0);
        }
        for (int i = 0; i < gamemap.GetButtonnum(); i++)
        {
            if (!press)
                press = gamemap.GetButton()[i]->Onpress(((player1.GetX2() + player1.GetX1()) / 2), player1.GetY2() - 1);
            if (!press)
                press = gamemap.GetButton()[i]->Onpress(((player2.GetX2() + player2.GetX1()) / 2), player2.GetY2() - 1);
        }
        for (int i = 0; i < (int)gamemap.GetRedButton().size(); i++)
        {
            if (!Redpress)
                Redpress = gamemap.GetRedButton()[i]->Onpress(((player1.GetX2()+player1.GetX1())/2), player1.GetY2() - 1);
            if (!Redpress)
                Redpress = gamemap.GetRedButton()[i]->Onpress(((player2.GetX2()+player2.GetX1())/2), player2.GetY2() - 1);
        }
        for (int i = 0; i < (int)gamemap.GetFallWall().size(); i++)
        {
            if (gamemap.GetFallWall()[i]->GetFall() == 0)       gamemap.GetFallWall()[i]->SetFall(player1);
            if (gamemap.GetFallWall()[i]->GetFall() == 0)       gamemap.GetFallWall()[i]->SetFall(player2);
        }
        for (int i = 0; i < (int)gamemap.GetChainElevator().size(); i++)
        {
            gamemap.GetChainElevator()[i]->SetFall(player1);
            gamemap.GetChainElevator()[i]->SetFall(player2);
            gamemap.GetChainElevator()[i]->SetFall2(player1);
            gamemap.GetChainElevator()[i]->SetFall2(player2);
```

```
		}
		for (int i = 0; i < (int)gamemap.GetRising().size(); i++)
		{
			if (!player1.Getfly())player1.Setfly(gamemap.GetRising()[i]->SetRising(&player1));
			if (!player2.Getfly())player2.Setfly(gamemap.GetRising()[i]->SetRising(&player2));
		}
		for (int i = 0; i < (int)gamemap.GetFallWall().size(); i++) gamemap.FallWallMove(i);
		gamemap.StoneDown();
		for (int i = 0; i < (int)gamemap.GetChainElevator().size(); i++) gamemap.ChainElevatorMove(i);
		int move = gamemap.elevatormove(press);
		if ((int)gamemap.GetRedButton().size() != 0) gamemap.makewall(Redpress);
		////////////////////////////////////////判斷四周是否為牆壁////////////////////////////////////
		player1.sethead(gamemap.wall(player1));
		player1.setleft(gamemap.leftwall(player1));
		player1.setright(gamemap.rightwall(&player1));
		player2.sethead(gamemap.wall(player2));
		player2.setleft(gamemap.leftwall(player2));
		player2.setright(gamemap.rightwall(&player2));
		player1.setfloor(gamemap.floor(&player1, move));
		player2.setfloor(gamemap.floor(&player2, move));
		player1.OnMove();
		player2.OnMove();
		////////////////////////////////////////人物死亡////////////////////////////////////////
		if (gamemap.gameover(player1))
		{
			CAudio::Instance()->Play(hit, false);
			CAudio::Instance()->Stop(gamemusic);
			GotoGameState(GAME_STATE_OVER);
		}
		if (gamemap.gameover(player2))
		{
			CAudio::Instance()->Play(hit, false);
			CAudio::Instance()->Stop(gamemusic);
			GotoGameState(GAME_STATE_OVER);
		}
		////////////////////////////////////////獲得寶石音效////////////////////////////////////////
		if (gamemap.isgem(player1)) { player1.Addgem(); CAudio::Instance()->Play(Gem, false); }
		if (gamemap.isgem(player2)) { player2.Addgem(); CAudio::Instance()->Play(Gem, false); }
		////////////////////////////////////////判斷各人物是否通關////////////////////////////////////
		winfire = gamemap.Clearance(player2);
		winice = gamemap.Clearance(player1);
		////////////////////////////////////////通關////////////////////////////////////////
		if (winfire&&winice)
		{
			stage[current_level - 1] = true;
			current_rank = true;
			CAudio::Instance()->Stop(gamemusic);
			CAudio::Instance()->Play(gamewin);
			GotoGameState(GAME_STATE_OVER);
		}
		else CSpecialEffect::SetCurrentTime();		//計時
	}
}
void CGameStateRun::OnInit()  // 遊戲的初值及圖形設定
{
	ShowInitProgress(33);	// 接個前一個狀態的進度，此處進度視為33%
	/////////////讀取圖片/////////////
	player1.LoadBitmap();
	player2.LoadBitmap();
	gamemap.LoadBitmap();
	showmovie.LoadBitmap();
	showmovie.Init();
	////////////////////////////////
	ShowInitProgress(50);	// 此處進度視為50%
}
void CGameStateRun::OnKeyDown(UINT nChar, UINT nRepCnt, UINT nFlags)
{
	const char KEY_SPACE = ' ';
	const char KEY_LEFT  = 0x25; // keyboard 左箭頭
	const char KEY_UP    = 0x26; // keyboard 上箭頭
	const char KEY_RIGHT = 0x27; // keyboard 右箭頭
	const char KEY_DOWN  = 0x28; // keyboard 下箭頭
	const char KEY_LEFT2 = 0x41; // keyboard A
	const char KEY_UP2 = 0x57; // keyboard W
	const char KEY_RIGHT2 = 0x44; // keyboard D
	const char KEY_DOWN2= 0x53; // keyboard S
```

```
	if (nChar == KEY_LEFT)
		player1.SetMovingLeft(true);
	if (nChar == KEY_RIGHT)
	  player1.SetMovingRight(true);
	if (nChar == KEY_UP)
	  player1.SetMovingUp(true);
	if (nChar == KEY_DOWN)
	  player1.SetMovingDown(true);
	if (nChar == KEY_LEFT2)
		player2.SetMovingLeft(true);
	if (nChar == KEY_RIGHT2)
		player2.SetMovingRight(true);
	if (nChar == KEY_UP2)
		player2.SetMovingUp(true);
	if (nChar == KEY_DOWN2)
		player2.SetMovingDown(true);
	if (nChar == KEY_SPACE)
		GotoGameState(GAME_STATE_RUN);
}
void CGameStateRun::OnKeyUp(UINT nChar, UINT nRepCnt, UINT nFlags)
{
	const char KEY_LEFT  = 0x25; // keyboard 左箭頭
	const char KEY_UP    = 0x26; // keyboard 上箭頭
	const char KEY_RIGHT = 0x27; // keyboard 右箭頭
	const char KEY_DOWN  = 0x28; // keyboard 下箭頭
	const char KEY_LEFT2 = 0x41; // keyboard A
	const char KEY_UP2 = 0x57; // keyboard W
	const char KEY_RIGHT2 = 0x44; // keyboard D
	const char KEY_DOWN2 = 0x53; // keyboard S
	if (nChar == KEY_LEFT)
		player1.SetMovingLeft(false);
	if (nChar == KEY_RIGHT)
	  player1.SetMovingRight(false);
	if (nChar == KEY_UP)
	  player1.SetMovingUp(false);
	if (nChar == KEY_DOWN)
	  player1.SetMovingDown(false);
	if (nChar == KEY_LEFT2)
		player2.SetMovingLeft(false);
	if (nChar == KEY_RIGHT2)
		player2.SetMovingRight(false);
	if (nChar == KEY_UP2)
		player2.SetMovingUp(false);
	if (nChar == KEY_DOWN2)
		player2.SetMovingDown(false);
}
void CGameStateRun::OnLButtonDown(UINT nFlags, CPoint point)  // 處理滑鼠的動作
{
	if (!showmovie.isOK())
	{
		if (((point.x > 235 && point.x < 423) && (point.y > 174 && point.y < 200)))
		{
			showmovie.ResetPos();
			showmovie.Init();
			GotoGameState(GAME_STATE_RUN);
		}
		else if (((point.x > 244 && point.x < 422) && (point.y > 285 && point.y < 310)))
		{
			showmovie.ResetPos();
			showmovie.Init();
			CAudio::Instance()->Stop(gamemusic);
			GotoGameState(GAME_STATE_INIT);
		}
	}
}
void CGameStateRun::OnLButtonUp(UINT nFlags, CPoint point) // 處理滑鼠的動作
{
}
void CGameStateRun::OnMouseMove(UINT nFlags, CPoint point)		// 處理滑鼠的動作
{
	if (((point.x > 310 && point.x < 380) && (point.y > 450 && point.y < 480))) showmovie.SetUp();
	if (point.x < 160 || point.x > 528) showmovie.SetDown();
	if (!showmovie.isOK())
	{
		if (((point.x > 235 && point.x < 423) && (point.y > 174 && point.y < 200)))		showmovie.GoRetry();
		else if (((point.x > 244 && point.x < 422) && (point.y > 285 && point.y < 310))) showmovie.GoMenu();
```

```
                else showmovie.GoInit();
        }
}
void CGameStateRun::OnShow()
{
        ///////////顯示地圖、角色、暫停畫面///////////////
        gamemap.OnShow();
        player2.OnShow();
        player1.OnShow();
        showmovie.OnShow();
        ////////////////////////////////////////////////
}
CGameStateAttack::CGameStateAttack(CGame * g)
        : CGameState(g)
{
}
CGameStateAttack::~CGameStateAttack()
{
        ////////////////////////////刪除剩餘子彈//////////////////////////
        player1.DeleteBullet();
        player2.DeleteBullet();
        ////////////////////////////////////////////////////////////////
}
void CGameStateAttack::OnBeginState()
{
        ////////////////////////人物初始化////////////////////////////////
        player1.Initialize();
        player2.Initialize();
        ////////////////////////地圖初始化////////////////////////////////
        gamemap.deleteObject();
        gamemap.Initialize(current_level, &player1, &player2);
        gamemap.LoadObject();
        ////////////////////////風向初始化////////////////////////////////
        wind.Init();
        ////////////////////////播放開頭音效及 BGM//////////////////////
        CAudio::Instance()->Play(Round1);
        CAudio::Instance()->Play(attackmusic);
        ////////////////////////開場動畫位置////////////////////////////
        Start.SetIceXY(player1.GetX1(), player1.GetY1());
        Start.SetFireXY(player2.GetX1(), player2.GetY1());
        Start.Restart();
        ////////////////////////刪除上場剩餘子彈////////////////////////
        player1.DeleteBullet();
        player2.DeleteBullet();
}
void CGameStateAttack::OnInit()
{
        ////////////////////////////////////讀取圖片及設定圖片初始位置////////////////////////////
        player1.LoadBitmap(1);
        player2.LoadBitmap(1);
        Start.LoadBitmap();
        background.LoadBitmap(pic_background);
        gamemap.LoadBitmap();
        background.SetTopLeft(0, 0);
        player1image.LoadBitmap(ICEATTACK2, RGB(0, 0, 0));
        player2image.LoadBitmap(FIRE_ATTTACK, RGB(0, 0, 0));
        player1image.SetTopLeft(574, 10);
        player2image.SetTopLeft(20, 10);
        fireballmovie.LoadBitmap();
        iceballmovie.LoadBitmap();
        wind.LoadBitmap();
        ////////////////////////////////////////////////////////////////////////////////////////
}
void CGameStateAttack::OnKeyDown(UINT nChar, UINT nRepCnt, UINT nFlags)
{
        const char KEY_SPACE = ' ';
        const char KEY_LEFT = 0x25; // keyboard 左箭頭
        const char KEY_UP = 0x26; // keyboard 上箭頭
        const char KEY_RIGHT = 0x27; // keyboard 右箭頭
        const char KEY_DOWN = 0x28; // keyboard 下箭頭
        const char KEY_LEFT2 = 0x41; // keyboard A
        const char KEY_UP2 = 0x57; // keyboard W
        const char KEY_RIGHT2 = 0x44; // keyboard D
        const char KEY_DOWN2 = 0x53; // keyboard S
        const char KEY_Z = 0x5a;
        const char KEY_J = 0x4a;
```

```
	const char KEY_1 = 0x61;
	const char KEY_2 = 0x62;
	const char KEY_K = 'K';
	if (nChar == KEY_LEFT)
	{
		player1.SetMovingLeft(true);
		player1.SetCombodelay();		//將 combo 歸零倒數計時設為 5
	}
	if (nChar == KEY_RIGHT)
	{
		player1.SetMovingRight(true);
		player1.SetCombodelay();
	}

	if (nChar == KEY_UP)
	{
		player1.SetMovingUp(true);
		player1.SetCombodelay();
	}

	if (nChar == KEY_DOWN)
	{
		player1.SetMovingDown(true);
		player1.SetCombodelay();
	}

	if (nChar == KEY_1)
	{
		player1.SetAttack(true);
		player1.SetCombodelay();
	}
	if (nChar == KEY_2)
	{
		player1.SetisMagic(true);
		player1.SetCombodelay();
		CAudio::Instance()->Play(13, true);  //播放集氣音效
	}
	if (nChar == KEY_LEFT2)
	{
		player2.SetMovingLeft(true);
		player2.SetCombodelay();
	}

	if (nChar == KEY_RIGHT2)
	{
		player2.SetMovingRight(true);
		player2.SetCombodelay();
	}

	if (nChar == KEY_UP2)
	{
		player2.SetMovingUp(true);
		player2.SetCombodelay();
	}

	if (nChar == KEY_DOWN2)
	{
		player2.SetMovingDown(true);
		player2.SetCombodelay();
	}

	if (nChar == KEY_J)
	{
		player2.SetAttack(true);
		player2.SetCombodelay();
	}

	if (nChar == KEY_K)
	{
		player2.SetisMagic(true);
		CAudio::Instance()->Play(12, true);    //播放集氣音效
	}
}
void CGameStateAttack::OnKeyUp(UINT nChar, UINT nRepCnt, UINT nFlags)
{
	const char KEY_LEFT = 0x25; // keyboard 左箭頭
```

```
	const char KEY_UP = 0x26; // keyboard 上箭頭
	const char KEY_RIGHT = 0x27; // keyboard 右箭頭
	const char KEY_DOWN = 0x28; // keyboard 下箭頭
	const char KEY_LEFT2 = 0x41; // keyboard A
	const char KEY_UP2 = 0x57; // keyboard W
	const char KEY_RIGHT2 = 0x44; // keyboard D
	const char KEY_DOWN2 = 0x53; // keyboard S
	const char KEY_Z = 0x5a;
	const char KEY_J = 0x4a;
	const char KEY_SPACE = ' ';
	const char KEY_2 = 0x62;
	const char KEY_K = 'K';
	if (nChar == KEY_LEFT)
		player1.SetMovingLeft(false);

	if (nChar == KEY_RIGHT)
		player1.SetMovingRight(false);

	if (nChar == KEY_UP)
		player1.SetMovingUp(false);

	if (nChar == KEY_DOWN)
		player1.SetMovingDown(false);

	if (nChar == KEY_2)
	{
		player1.SetisMagic(false);
		CAudio::Instance()->Stop(13);		//暫停集氣音效
	}

	if (nChar == KEY_LEFT2)
		player2.SetMovingLeft(false);

	if (nChar == KEY_RIGHT2)
		player2.SetMovingRight(false);

	if (nChar == KEY_UP2)
		player2.SetMovingUp(false);

	if (nChar == KEY_DOWN2)
		player2.SetMovingDown(false);

	if (nChar == KEY_K)
	{
		player2.SetisMagic(false);
		CAudio::Instance()->Stop(12);		//暫停集氣音效
	}
}
void CGameStateAttack::OnMove()
{
	//////////////////開場動畫//////////////////
	if (!Start.Getend())
	{
		Start.OnMove();
	}
	//////////////////技能動畫//////////////////
	else if (fireballmovie.isStart())
	{
		fireballmovie.OnMove();
	}
	else if (iceballmovie.isStart())
	{
		iceballmovie.OnMove();
	}
	//////////////////遊戲運行//////////////////
	else
	{
		////////////////////////////////////下落電梯判斷////////////////////////////////////////////////
		for (int i = 0; i < (int)gamemap.GetFallWall().size(); i++) gamemap.GetFallWall()[i]->SetFall(0);
		for (int i = 0; i < (int)gamemap.GetFallWall().size(); i++)
		{
			if (gamemap.GetFallWall()[i]->GetFall() == 0)		gamemap.GetFallWall()[i]->SetFall(player1);
			if (gamemap.GetFallWall()[i]->GetFall() == 0)		gamemap.GetFallWall()[i]->SetFall(player2);
		}
		for (int i = 0; i < (int)gamemap.GetFallWall().size(); i++) gamemap.FallWallMove(i);
		////////////////////////////////////二段跳刷新////////////////////////////////////////////////
```

```
		if (player1.GetY1() == player1.Getfloor() - 39)
			player1.SetTwojump(true);
		if (player2.GetY1() == player2.Getfloor() - 39)
			player2.SetTwojump(true);
		////////////////////////////////判斷角色四周//////////////////////////////////////////////
		player1.sethead(gamemap.wall(player1));
		player1.setleft(gamemap.leftwall(player1));
		player1.setright(gamemap.rightwall(&player1));
		player2.sethead(gamemap.wall(player2));
		player2.setleft(gamemap.leftwall(player2));
		player2.setright(gamemap.rightwall(&player2));
		player1.setfloor(gamemap.floor(&player1));
		player2.setfloor(gamemap.floor(&player2));
		player1.OnMove(&player2);
		player2.OnMove(&player1);
		////////////////////////////////風向移動角色及計時//////////////////////////////////////////////
		wind.onMove(&player1);
		wind.onMove(&player2);
		wind.Count();
		////////////////////////////////技能動畫//////////////////////////////////////////////////////
		if (player1.SetShooting())
			iceballmovie.Init();
		if (player2.SetShooting())
			fireballmovie.Init();
		////////////////////////////////衝刺技能判斷//////////////////////////////////////////////////
		player1.SetRush();
		player2.SetRush();
		////////////////////////////////遊戲結束//////////////////////////////////////////////////////
		if (gamemap.gameover(player1))
		{
			CAudio::Instance()->Play(hit, false);
			CAudio::Instance()->Stop(attackmusic);
			winner = 1;
			GotoGameState(GAME_STATE_OVER);
		}
		if (gamemap.gameover(player2))
		{
			CAudio::Instance()->Play(hit, false);
			CAudio::Instance()->Stop(attackmusic);
			winner = 2;
			GotoGameState(GAME_STATE_OVER);
		}
	}
}
void CGameStateAttack::OnShow()
{
	////////////////////////////////////顯示地圖/////////////////////////////
	gamemap.OnShow();
	////////////////////////////判斷是否顯示開場動畫//////////////////////
	if (!Start.Getend()) {Start.OnShow();}
	else
	{
		////////////////////////顯示角色/////////////////////////////////
		player1image.ShowBitmap(1.5);
		player2image.ShowBitmap(1.5);
		player2.OnShow(1);
		player1.OnShow(1);
		////////////////////////顯示技能動畫/////////////////////////////////
		if (fireballmovie.isStart())
			fireballmovie.OnShow();
		else if (iceballmovie.isStart())
			iceballmovie.OnShow();
		////////////////////////顯示風向/////////////////////////////////////
		wind.OnShow();
	}
}
CGameStateNaruto::CGameStateNaruto(CGame * g): CGameState(g)
{
}
CGameStateNaruto::~CGameStateNaruto()
{
	////////////////////////////刪除人物物件/////////////////////////////////////
	player1.Deleteobject();
	delete player2;
}
void CGameStateNaruto::OnBeginState()
```

```
{
	//////////////////////////////////////初始地圖//////////////////////////////////////
	background.SetTopLeft(0, 0);
	Skillbackground.SetTopLeft(0, 0);
	//////////////////////////////////////判斷第幾關敵人//////////////////////////////////
	if (stage1) { delete player2; player2 = new Shikamaru; }
	if (stage2) { delete player2; player2 = new Neji;}
	if (stage3) { delete player2; player2 = new Orochimaru; }
	//////////////////////////////////////人物初始化//////////////////////////////////////
	player1.Initialize();
	player2->LoadBitmap();
	player2->SetPlayer(&player1);
	if(!stage3)CAudio::Instance()->Play(NarutoBGM,true);
	else CAudio::Instance()->Play(NarutoBGM2, true);
	if(stage1) kakainstrctor.ChangeSection(1);
}
void CGameStateNaruto::OnInit()
{
	//////////////////////////////////////UI 圖初始化//////////////////////////////////////
	Player1UI.Init(20, 20, "player", 128, SASUKE, RGB(73, 176, 255));
	Player2UI.Init(426, 20, "AI", 128, ShikamaruUI, RGB(255, 100, 100));
	Player3UI.Init(426, 20, "AI", 128, NejiUI, RGB(0, 128, 128));
	Player4UI.Init(426, 20, "AI", 128, OrochimaruUI, RGB(0, 128, 128));
	//////////////////////////////////////各圖片讀取//////////////////////////////////////
	player1.LoadBitmap();
	background.LoadBitmap(NarutoBackground);
	Skillbackground.AddBitmap(SkillBackground1);
	for(int i=0;i<4;i++) Skillbackground.AddBitmap(SkillBackground2+i);
	Skillbackground.SetDelayCount(3);
	Skillbackground.Reset();
	kakainstrctor.Init();
	//////////////////////////////////////關卡初始化//////////////////////////////////////
	stage1 = true;
	stage2 = stage3=false;
}
void CGameStateNaruto::OnKeyDown(UINT nChar, UINT nRepCnt, UINT nFlags)
{
	const char KEY_LEFT = 0x25; // keyboard 左箭頭
	const char KEY_UP = 0x5a; // keyboard 上箭頭
	const char KEY_RIGHT = 0x27; // keyboard 右箭頭
	const char KEY_DOWN = 0x28; // keyboard 下箭頭
	const char KEY_X = 0x58;
	const char KEY_A = 0x41;
	const char KEY_B = 0x42;
	if (kakainstrctor.IsDone())
	{
		if (nChar == KEY_LEFT)
			player1.SetMovingLeft(true);
		if (nChar == KEY_RIGHT)
			player1.SetMovingRight(true);
		if (nChar == KEY_UP)
			player1.SetMovingUp(true);
		if (nChar == KEY_X && player1.GetMove())
		{
			player1.Addattacknum();
			player1.SetisMolding(true);
		}
		if (nChar == ' ')
			player1.ChangeChakura(128);
		if (nChar == KEY_A)
			player2->SetSkill();
		if (nChar == KEY_B)
			player2->SetSkill2();
	}
}
void CGameStateNaruto::OnKeyUp(UINT nChar, UINT nRepCnt, UINT nFlags)
{
	const char KEY_LEFT = 0x25; // keyboard 左箭頭
	const char KEY_UP = 0x5a; // keyboard 上箭頭
	const char KEY_RIGHT = 0x27; // keyboard 右箭頭
	const char KEY_DOWN = 0x28; // keyboard 下箭頭
	const char KEY_X = 0x58;
	const char KEY_S = 0x53;
	const char KEY_Z = 0x5a;

	if (kakainstrctor.IsDone())
```

```
	{
		if (nChar == KEY_LEFT)
			player1.SetMovingLeft(false);
		if (nChar == KEY_RIGHT)
			player1.SetMovingRight(false);

		if (nChar == KEY_X)
			player1.SetisMolding(false);
	}
	else
	{
		if (nChar == KEY_S)
		{
			if (kakainstrctor.GetSection() == 5 || (kakainstrctor.GetSection() == 3))
			{
				CAudio::Instance()->Stop(NarutoBGM);
				GotoGameState(GAME_STATE_INIT);
			}
			kakainstrctor.SetDone(true);
		}
		if (nChar == KEY_Z && kakainstrctor.GetEnd())
			if(kakainstrctor.GetSection() == 1)kakainstrctor.ChangeFilm(kakainstrctor.GetFilm() + 1);
			else if (kakainstrctor.GetSection() == 2)kakainstrctor.ChangeVicFilm(kakainstrctor.GetVicFilm() + 1);
			else if (kakainstrctor.GetSection() == 4)kakainstrctor.ChangeVic2Film(kakainstrctor.GetVic2Film() + 1);
			else if (kakainstrctor.GetSection() == 5 || (kakainstrctor.GetSection() == 3))
			{
				CAudio::Instance()->Stop(NarutoBGM);
				GotoGameState(GAME_STATE_INIT);
			}
		if (nChar == KEY_Z && kakainstrctor.IsFinish())
		{
			if (kakainstrctor.GetSection() == 5 || (kakainstrctor.GetSection() == 3))
			{
				CAudio::Instance()->Stop(NarutoBGM);
				GotoGameState(GAME_STATE_INIT);
			}
			kakainstrctor.SetDone(true);
		}
	}
}
//////////////////////////////////////////////////////////////判斷碰撞//////////////////////////////////////////////////////////////
bool CGameStateNaruto::Hit()
{
	if (player1.GetX1() >= player2->GetX1() && player1.GetX1() <= player2->GetX2() && player1.GetY1() >= player2->GetY1() &&
	player1.GetY1() <= player2->GetY2())
		return true;
	if (player1.GetX2() >= player2->GetX1() && player1.GetX2() <= player2->GetX2() && player1.GetY1() >= player2->GetY1() &&
	player1.GetY1() <= player2->GetY2())
		return true;
	if (player1.GetX1() >= player2->GetX1() && player1.GetX1() <= player2->GetX2() && player1.GetY2() >= player2->GetY1() &&
	player1.GetY2() <= player2->GetY2())
		return true;
	if (player1.GetX2() >= player2->GetX1() && player1.GetX2() <= player2->GetX2() && player1.GetY2() >= player2->GetY1() &&
	player1.GetY2() <= player2->GetY2())
		return true;
	if (player2->GetX1() >= player1.GetX1() && player2->GetX1() <= player1.GetX2() && player2->GetY1() >= player1.GetY1() &&
	player2->GetY1() <= player1.GetY2())
		return true;
	if (player2->GetX2() >= player1.GetX1() && player2->GetX2() <= player1.GetX2() && player2->GetY1() >= player1.GetY1() &&
	player2->GetY1() <= player1.GetY2())
		return true;
	if (player2->GetX1() >= player1.GetX1() && player2->GetX1() <= player1.GetX2() && player2->GetY2() >= player1.GetY1() &&
	player2->GetY2() <= player1.GetY2())
		return true;
	if (player2->GetX2() >= player1.GetX1() && player2->GetX2() <= player1.GetX2() && player2->GetY2() >= player1.GetY1() &&
	player2->GetY2() <= player1.GetY2())
		return true;
	return false;
}
bool CGameStateNaruto::ShootHit(Sasuke _player1, Shoot *shoot)
{
	if (_player1.GetX1() >= shoot->GetX1() && _player1.GetX1() <= shoot->GetX2() && _player1.GetY1() >= shoot->GetY1() &&
	_player1.GetY1() <= shoot->GetY2())
		return true;
	if (_player1.GetX2() >= shoot->GetX1() && _player1.GetX2() <= shoot->GetX2() && _player1.GetY1() >= shoot->GetY1() &&
	_player1.GetY1() <= shoot->GetY2())
```

```
		return true;
	if (_player1.GetX1() >= shoot->GetX1() && _player1.GetX1() <= shoot->GetX2() && _player1.GetY2() >= shoot->GetY1() &&
	_player1.GetY2() <= shoot->GetY2())
		return true;
	if (_player1.GetX2() >= shoot->GetX1() && _player1.GetX2() <= shoot->GetX2() && _player1.GetY2() >= shoot->GetY1() &&
	_player1.GetY2() <= shoot->GetY2())
		return true;
	if (shoot->GetX1() >= _player1.GetX1() && shoot->GetX1() <= _player1.GetX2() && shoot->GetY1() >= _player1.GetY1() &&
	shoot->GetY1() <= _player1.GetY2())
		return true;
	if (shoot->GetX2() >= _player1.GetX1() && shoot->GetX2() <= _player1.GetX2() && shoot->GetY1() >= _player1.GetY1() &&
	shoot->GetY1() <= _player1.GetY2())
		return true;
	if (shoot->GetX1() >= _player1.GetX1() && shoot->GetX1() <= _player1.GetX2() && shoot->GetY2() >= _player1.GetY1() &&
	shoot->GetY2() <= _player1.GetY2())
		return true;
	if (shoot->GetX2() >= _player1.GetX1() && shoot->GetX2() <= _player1.GetX2() && shoot->GetY2() >= _player1.GetY1() &&
	shoot->GetY2() <= _player1.GetY2())
		return true;
	return false;
}
bool CGameStateNaruto::ShootHit(EnemyAI * _player1, Shoot * shoot)
{
	if (_player1->GetX1() >= shoot->GetX1() && _player1->GetX1() <= shoot->GetX2() && _player1->GetY1() >= shoot->GetY1() &&
	_player1->GetY1() <= shoot->GetY2())
		return true;
	if (_player1->GetX2() >= shoot->GetX1() && _player1->GetX2() <= shoot->GetX2() && _player1->GetY1() >= shoot->GetY1() &&
	_player1->GetY1() <= shoot->GetY2())
		return true;
	if (_player1->GetX1() >= shoot->GetX1() && _player1->GetX1() <= shoot->GetX2() && _player1->GetY2() >= shoot->GetY1() &&
	_player1->GetY2() <= shoot->GetY2())
		return true;
	if (_player1->GetX2() >= shoot->GetX1() && _player1->GetX2() <= shoot->GetX2() && _player1->GetY2() >= shoot->GetY1() &&
	_player1->GetY2() <= shoot->GetY2())
		return true;
	if (shoot->GetX1() >= _player1->GetX1() && shoot->GetX1() <= _player1->GetX2() && shoot->GetY1() >= _player1->GetY1() &&
	shoot->GetY1() <= _player1->GetY2())
		return true;
	if (shoot->GetX2() >= _player1->GetX1() && shoot->GetX2() <= _player1->GetX2() && shoot->GetY1() >= _player1->GetY1() &&
	shoot->GetY1() <= _player1->GetY2())
		return true;
	if (shoot->GetX1() >= _player1->GetX1() && shoot->GetX1() <= _player1->GetX2() && shoot->GetY2() >= _player1->GetY1() &&
	shoot->GetY2() <= _player1->GetY2())
		return true;
	if (shoot->GetX2() >= _player1->GetX1() && shoot->GetX2() <= _player1->GetX2() && shoot->GetY2() >= _player1->GetY1() &&
	shoot->GetY2() <= _player1->GetY2())
		return true;
	return false;
}
//////////////////////////////////////////////////////////////////////////////////////////////////////////////////////
void CGameStateNaruto::OnMove()
{
	if (kakainstrctor.IsDone())
	{
		//////////////////////////////////////////////////////人物移動//////////////////////////////////////////
		player1.OnMove();
		player2->UI(Hit());
		//////////////////////////////////////////////攻擊判定//////////////////////////////////////////////
		if ((player1.GetCurrentBitmapNumber() == 40 || player1.GetCurrentBitmapNumber() == 49) && Hit()
		&& !player2->GetDie())
		{
			player2->SetAttacked(true);
			player2->SetDirection(0);
		}
		else if ((player1.GetCurrentBitmapNumber() == 180 || player1.GetCurrentBitmapNumber() == 189) &&
		Hit() && !player2->GetDie())
		{
			player2->SetAttacked(true);
			player2->SetDirection(1);
		}
		else if ((player1.GetCurrentBitmapNumber() == 61 || player1.GetCurrentBitmapNumber() == 27) &&
		Hit() && !player2->GetDie())
			player2->SetDie(0);
		else if ((player1.GetCurrentBitmapNumber() == 201 || player1.GetCurrentBitmapNumber() == 167) &&
		Hit() && !player2->GetDie())
			player2->SetDie(1);
```

```
//////////////////////////////////////////////技能判定//////////////////////////////////////////////////////////
if (player1.GetisSkill() && Hit() && (player1.GetSkillCurrentBitmapNumber() == 14 + player1.GetDirection() * 44 ||
player1.GetSkillCurrentBitmapNumber() == 23 + player1.GetDirection() * 44 ||
player1.GetSkillCurrentBitmapNumber() == 32 + player1.GetDirection() * 44 ||
player1.GetSkillCurrentBitmapNumber() == 41 + player1.GetDirection() * 44)) player2->SetAttacked(true);
//////////////////////////////////////////////飛行道具判定//////////////////////////////////////////////////////////
for (int i = 0; i < (int)player2->GetShurikan().size(); i++)
	if (ShootHit(player1, player2->GetShurikan()[i]))
	{
		player1.SetAttacked(true);
		player2->SetShurikanDelete(i);
	}
for (int i = 0; i < (int)player1.GetFireball().size(); i++)
	if (ShootHit(player2, player1.GetFireball()[i]))
	{
		if (player2->GetMove()) { player2->SetAttacked(true); }
		player1.SetFireballDelete(i);
	}
for (int i = 0; i < (int)player1.GetShurikan().size(); i++)
	if (ShootHit(player2, player1.GetShurikan()[i]))
	{
		if (player2->GetMove()) { player2->SetAttacked(true); }
		player1.SetShurikanDelete(i);
	}
//////////////////////////////////////////////敵人移動//////////////////////////////////////////////////////////
player2->OnMove();
//////////////////////////////////////////////判斷敵人是否死亡//////////////////////////////////////////////////////////
if (player2->Getblood() <= 0)
{
	player1.Deleteobject();
	if (stage1)
	{
		stage1 = false;
		stage2 = true;
		kakainstrctor.ChangeSection(2);
		GotoGameState(GAME_STATE_NARUTO);
	}
	else if (stage2)
	{
		kakainstrctor.ChangeSection(4);
		stage3 = true;
		stage2 = false;
		GotoGameState(GAME_STATE_NARUTO);
	}
	else if (stage3)
	{
		stage1 = true;
		stage3 = false;
		CAudio::Instance()->Stop(NarutoBGM2);
		kakainstrctor.ChangeSection(5);
	}
}
//////////////////////////////////////////////技能動畫//////////////////////////////////////////////////////////
if (!player2->GetisSkill() && player2->GetMove() && player1.GetisSkill() && Hit() &&
player1.GetSkillCurrentBitmapNumber() >= 114 + player1.GetDirection() * 24 &&
player1.GetSkillCurrentBitmapNumber() <= 137 + player1.GetDirection() * 24 && !player1.GetSkillsuccess())
{
	player1.SetSkillsuccess();
	CAudio::Instance()->Play(28);
	player1.GetSkillsuccessMovie()->Movetonum(0 + player1.GetDirection() * 53);
}
if (player1.GetSkillsuccessMovie()->GetCurrentBitmapNumber() >= 4 + player1.GetDirection() * 53 &&
player1.GetSkillsuccessMovie()->GetCurrentBitmapNumber() <= 37 + player1.GetDirection() * 53)
{
	player2->SetAttacked(true);
	player2->Setblood(player2->Getblood()+1);
}
else if (player1.GetSkillsuccessMovie()->GetCurrentBitmapNumber() == 52 + player1.GetDirection() * 53)
{
	player2->SetDie(true);
	player2->SetonFloor();
}
//////////////////////////////////////////////判斷技能是否命中//////////////////////////////////////////////////////////
if (player1.GetSkillsuccess())
{
	player2->SetMove(false);
```

```
                if (player1.GetDirection() == 0)player2->SetXY(player1.GetX2(), player1.GetY1()+20);
                else player2->SetXY(player1.GetX1() - 50, player1.GetY1()+20);
        }
        ////////////////////////////////////////////////各人物 UI 更新//////////////////////////////////////////////////////////
        Player1UI.ChangeChakura(player1.GetChakura());
        Player1UI.ChangeBlood(player1.Getblood());
        Player2UI.ChangeBlood(player2->Getblood());
        Player3UI.ChangeBlood(player2->Getblood());
        Player4UI.ChangeBlood(player2->Getblood());
        Player1UI.SetisMolding(player1.GetisMolding(), player1.GetMagic());
        ////////////////////////////////////////////////判斷敵方是否使用技能//////////////////////////////////////////////////////
        if (player1.Getskillstart()->isStart()) player2->SetMove(false);
        else if (player1.GetisSkill()) player2->SetMove(true);
        ////////////////////////////////////////////////更換技能背景//////////////////////////////////////////////////////////
        if (player1.GetSkillsuccess())Skillbackground.OnMove();
        ////////////////////////////////////////////////死亡//////////////////////////////////////////////////////////
        if (player1.Getblood() <= 0)
        {
                player1.Deleteobject();
                kakainstrctor.ChangeSection(3);
                CAudio::Instance()->Stop(NarutoBGM);
                CAudio::Instance()->Stop(NarutoBGM2);
                stage2 = false;
                stage3 = false;
                stage1 = true;
        }
    }
}
void CGameStateNaruto::OnShow()
{
    ////////////////////////////////////////////////顯示圖片//////////////////////////////////////////////
    background.ShowBitmap();
    if(player1.GetSkillsuccess()) Skillbackground.OnShow();
    player2->OnShow();
    player1.OnShow();
    Player1UI.OnShow();
    if (stage1)Player2UI.OnShow();
    if (stage2)Player3UI.OnShow();
    if (stage3)Player4UI.OnShow();
    if (player1.Getskillstart()->isStart()) player1.Getskillstart()->OnShow();
    if (player2->Getskillstart()->isStart()) player2->Getskillstart()->OnShow();
    kakainstrctor.OnShow(kakainstrctor.GetSection());
}
}
```

CgameMap.h

```
#include "yellowbutton.h"
#include "elevator.h"
#include "man.h"
#include "iceman.h"
#include "fireman.h"
#include "Stone.h"
#include "firedoor.h"
#include "icedoor.h"
#include "fire.h"
#include "water.h"
#include "poison.h"
#include "FallWall.h"
#include "RedButton.h"
#include "Rising.h"
#include "Chainelevator.h"
#include "lever.h"
namespace game_framework {
    class CgameMap {
    public:
        CgameMap();
        ~CgameMap();
        void Initialize(int i);
        void Initialize(int i,man *player1,man *player2);
        int GetButtonnum();
        vector<yellowbutton*> GetButton();
        vector<RedButton*> GetRedButton();
        vector<FallWall*> GetFallWall();
        vector<Rising*> GetRising();
        vector<ChainElevator*> GetChainElevator();
        void LoadBitmap();
```

```
		void LoadObject();
		void OnShow();
		int floor(man *player,int move);
		int floor(man *player);
		bool wall(man player);
		int leftwall(man player);
		int rightwall(man *player);
		int elevatormove(bool flag);
		void FallWallMove(int j);
		void ChainElevatorMove(int j);
		void PushStone(int direction);
		void StoneDown();
		bool gameover(man player);
		bool isgem(man player);
		bool Clearance(man player);
		int OnSlade(man *player, int i, int j);
		int OnRSlade(man *player, int i, int j);
		void makewall(bool flag);
		void LeverOn(bool flag);
		void deleteObject();
		//bool CheckPosition(int i, int j, man player);
	protected:
		CMovingBitmap down, slade,Rslade, icegem, firegem, rock, lr, dul, dur, durl,background;
		water _water;
		fire _fire;
		poison _poison;
		vector<RedButton*> Redbutton;
		vector<buttonelevator*> _elevator;
		vector<yellowbutton*> button;
		vector<rockStone*> stone;
		vector<FallWall*> fallwall;
		vector<Rising*> rising;
		vector<ChainElevator*> chainelevator;
		vector<lever*> Lever;
		int buttonnum,elevatornum,icegemnum,firegemnum,stonenum,fallwallnum, redbuttonnum,risingnum,chainelevatornum,Levernum;
		firedoor fdoor;
		icedoor idoor;
		int map[24][32];
		int delaycount;
		const int MW, MH;
	};
}
```

CgameMap.cpp

```
#include "stdafx.h"
#include "Resource.h"
#include <mmsystem.h>
#include <ddraw.h>
#include "audio.h"
#include "gamelib.h"
#include "CgameMap.h"
#include <fstream>
#include <string>
#include <strstream>
#include <iostream>
namespace game_framework {
	CgameMap::CgameMap()
	:MW(20), MH(20), buttonnum(0), elevatornum(0), icegemnum(0), firegemnum(0), stonenum(0), fallwallnum(0), risingnum(0),delaycount(5)
	{
		Initialize(1);
	}

	CgameMap::~CgameMap()
	{
		//////////////////////解構地圖物件//////////////////////////
		for (int i = 0; i < buttonnum; i++) delete button[i];
		for (int i = 0; i < buttonnum; i++) button.erase(button.begin());
		for (int i = 0; i < elevatornum; i++) delete _elevator[i];
		for (int i = 0; i < elevatornum; i++) _elevator.erase(_elevator.begin());
		for (int i = 0; i < stonenum; i++) delete stone[i];
		for (int i = 0; i < stonenum; i++) stone.erase(stone.begin());
		for (int i = 0; i < fallwallnum; i++) delete fallwall[i];
		for (int i = 0; i < fallwallnum; i++) fallwall.erase(fallwall.begin());
		for (int i = 0; i < redbuttonnum; i++) delete Redbutton[i];
		for (int i = 0; i < redbuttonnum; i++) Redbutton.erase(Redbutton.begin());
		for (int i = 0; i < risingnum; i++) delete rising[i];
```

```
	for (int i = 0; i < risingnum; i++) rising.erase(rising.begin());
	for (int i = 0; i < chainelevatornum; i++) delete chainelevator[i];
	for (int i = 0; i < chainelevatornum; i++) chainelevator.erase(chainelevator.begin());
	for (int i = 0; i < Levernum; i++) delete Lever[i];
	for (int i = 0; i < Levernum; i++) Lever.erase(Lever.begin());
	//////////////////////////////////////////////////////////////
}

void CgameMap::Initialize(int mapID)
{
	//////////////////////////////////////////////////////////初始化//////////////////////////////////////////
	buttonnum = 0, elevatornum = 0, icegemnum = 0, firegemnum = 0, stonenum = 0, fallwallnum = 0, redbuttonnum = 0, risingnum =
	0,chainelevatornum=0,Levernum=0;
	int count = 0, flag = 0;
	string MapId = "Map" + to_string(mapID), str;
	int map_init[24][32];
	ifstream inFile("maps.txt");
	//////////////////////////////////////////////////////////讀檔//////////////////////////////////////////
	while (!inFile.eof())
	{
		getline(inFile, str);
		if (str == MapId)
		{
			for (int i = 0; i < 24; i++)
				for (int j = 0; j < 32; j++)
					inFile >> map_init[i][j];
			break;
		}
	}

	for (int i = 0; i < 24; i++)
	{
		for (int j = 0; j < 32; j++)
		{
			map[i][j] = map_init[i][j];
			switch (map[i][j])
			{
			case 2:
				buttonnum++;
				button.push_back(new yellowbutton);
				break;
			case 3:
				count++;
				if (count == 3)
				{
					elevatornum++;
					_elevator.push_back(new buttonelevator);
				}
				break;
			case 7:
				firegemnum++;
				break;
			case 8:
				icegemnum++;
				break;
			case 11:
				stonenum++;
				stone.push_back(new rockStone);
				break;
			case 14:
				count++;
				if (count == 3)
				{
					fallwallnum++;
					fallwall.push_back(new FallWall);
				}
				break;
			case 15:
				redbuttonnum++;
				Redbutton.push_back(new RedButton);
				break;
			case 17:
				count++;
				if (count == 3)
				{
					risingnum++;
```

```
                                        rising.push_back(new Rising);
                                }
                                break;
                        case 18:
                                count++;
                                if (count == 3)
                                {
                                        flag++;
                                        count = 0;
                                }
                                if (flag == 2)
                                {
                                        chainelevatornum++;
                                        chainelevator.push_back(new ChainElevator);
                                }
                                break;
                        case 20:
                                Levernum++;
                                Lever.push_back(new lever);
                        default:
                                count = 0;
                                break;
                        }
                }
        }
        inFile.close();
        ////////////////////////////////////////////////////////////////////////////
}

void CgameMap::Initialize(int i, man* player1, man* player2)
{
        ////////////////////////////////////角色位置讀檔//////////////////////////////
        Initialize(i);
        for (int i = 0; i < 24; i++)
        {
                for (int j = 0; j < 32; j++)
                {
                        if (map[i][j] == 99) { player1->SetXY((MW*j), (MH*i)); map[i][j] = 0; }
                        if (map[i][j] == 98) { player2->SetXY((MW*j), (MH*i)); map[i][j] = 0; }
                }
        }
        ////////////////////////////////////////////////////////////////////////////
}

int CgameMap::GetButtonnum()
{
        return buttonnum;
}

vector<yellowbutton*> CgameMap::GetButton()
{
        return button;
}

vector<RedButton*> CgameMap::GetRedButton()
{
        return Redbutton;
}

vector<FallWall*> CgameMap::GetFallWall()
{
        return fallwall;
}

vector<Rising*> CgameMap::GetRising()
{
        return rising;
}

vector<ChainElevator*> CgameMap::GetChainElevator()
{
        return chainelevator;
}

void CgameMap::LoadBitmap()
{
```

```
	//////////////////////////////////////////////讀取圖片//////////////////////////////////////
	down.LoadBitmap(pic_d);
	slade.LoadBitmap(pic_slade, RGB(255, 255, 255));
	background.LoadBitmap(ground);
	Rslade.LoadBitmap(rslade, RGB(255, 255, 255));
	icegem.LoadBitmap(bluegem, RGB(255, 255, 255));
	firegem.LoadBitmap(redgem, RGB(255, 255, 255));
	idoor.LoadBitmap();
	fdoor.LoadBitmap();
	_water.LoadBitmap();
	_fire.LoadBitmap();
	_poison.LoadBitmap();
	//////////////////////////////////////////////////////////////////////////////////////////
}

void CgameMap::LoadObject()
{
	//////////////////////////////////物件讀取圖片//////////////////////////////////////////
	int elevatorcount = 0, stonecount = 0, fallwallcount = 0,risingcount=0,chainelevatorcount=0,flag=0;
	for (int i = 0; i < buttonnum; i++)	button[i]->LoadBitmap();
	for (int i = 0; i < stonenum; i++)	stone[i]->LoadBitmap();
	for (int i = 0; i < elevatornum; i++)	_elevator[i]->LoadBitmap();
	for (int i = 0; i < fallwallnum; i++)	fallwall[i]->LoadBitmap();
	for (int i = 0; i < risingnum; i++)	rising[i]->LoadBitmap();
	for (int i = 0; i < chainelevatornum; i++)	chainelevator[i]->LoadBitmap();
	for (int i = 0; i < Levernum; i++)	Lever[i]->LoadBitmap();
	//////////////////////////////////物件設定位置//////////////////////////////////////
	for (int i = 0; i < 24; i++)
	{
		for (int j = 0; j < 32; j++)
		{
			switch (map[i][j])
			{
			case 3:
				_elevator[elevatorcount]->setoxy((MW*j), (MH*i));
				_elevator[elevatorcount++]->SetTopLeft((MW*j), (MH*i));
				j = j + 2;
				break;
			case 11:
				stone[stonecount++]->SetXY((MW*j), (MH*i));
				break;
			case 14:
				fallwall[fallwallcount]->Setoxy((MW*j), (MH*i));
				fallwall[fallwallcount++]->SetTopLeft((MW*j), (MH*i));
				j = j + 2;
				break;
			case 17:
				rising[risingcount++]->SetXY((MW*j), (MH*i));
				j = j + 2;
				break;
			case 18:
				if (flag == 0)
				{
					chainelevator[chainelevatorcount]->SetTopLeft((MW*j), (MH*i));
					flag = 1;
					j = j + 2;
				}
				else if (flag == 1)
				{
					chainelevator[chainelevatorcount++]->SetX2Y2((MW*j),(MH*i));
					flag = 0;
					j = j + 2;
				}
				break;
			default:
				break;
			}
		}
		//////////////////////////////////////設定電梯範圍//////////////////////////
	}
	for (int i = 0; i < 24; i++)
	{
		for (int j = 0; j < 32; j++)
		{
			switch (map[i][j])
			{
```

```
                        case 19:
                                for (int k = 0; k < chainelevatorcount; k++)
                                {
                                        if ((MW*j) == chainelevator[k]->GetX())
                                                chainelevator[k]->Setaimy1((MH*i));
                                        else if (((MW*j) == chainelevator[k]->GetX2()))
                                                chainelevator[k]->Setaimy2((MH*i));
                                }
                                map[i][j] = 0;
                                break;
                        default:
                                break;
                        }
                }
        }
        ////////////////////////////////讀取按鈕圖片//////////////////////
        for (int i = 0; i < redbuttonnum; i++) Redbutton[i]->LoadBitmap();
        ////////////////////////////////設定陷阱流速//////////////////////
        _fire.SetDelayTime();
        _water.SetDelayTime();
        _poison.SetDelayTime();
        ////////////////////////////////////////////////////////////////
}

void CgameMap::OnShow()
{
        ////////////////////////////////根據地圖資訊建立地圖///////////////
        int snum = 0;
        int bnum = 0;
        int Rbnum = 0;
        int Lnum = 0;
        for (int i = 0; i < 24; i++)
                for (int j = 0; j < 32; j++)
                {
                        switch (map[i][j]) {
                        case 0:
                                background.SetTopLeft((MW*j), (MH*i));
                                background.ShowBitmap();
                                break;
                        case 1:
                                down.SetTopLeft((MW*j), (MH*i));
                                down.ShowBitmap();
                                break;
                        case 2:
                                background.SetTopLeft((MW*j), (MH*i));
                                background.ShowBitmap();
                                button[bnum]->SetTopLeft((MW*j), (MH*i));
                                button[bnum++]->ShowBitmap();
                                break;
                        case 3:
                                background.SetTopLeft((MW*j), (MH*i));
                                background.ShowBitmap();
                                break;
                        case 4:
                                down.SetTopLeft((MW*j), (MH*i));
                                down.ShowBitmap();
                                _fire.OnShow((MW*j), (MH*i));
                                break;
                        case 5:
                                down.SetTopLeft((MW*j), (MH*i));
                                down.ShowBitmap();
                                _water.OnShow((MW*j), (MH*i));
                                break;
                        case 6:
                                down.SetTopLeft((MW*j), (MH*i));
                                down.ShowBitmap();
                                _poison.OnShow((MW*j), (MH*i));
                                break;
                        case 7:
                                background.SetTopLeft((MW*j), (MH*i));
                                background.ShowBitmap();
                                firegem.SetTopLeft((MW*j), (MH*i)+5);
                                firegem.ShowBitmap();
                                break;
                        case 8:
                                background.SetTopLeft((MW*j), (MH*i));
```

```
                        background.ShowBitmap();
                        icegem.SetTopLeft((MW*j), (MH*i)+5);
                        icegem.ShowBitmap();
                        break;
                case 9:
                        j++;
                        background.SetTopLeft((MW*j), (MH*i));
                        background.ShowBitmap();
                        fdoor.OnShow((MW*j) - 40, (MH*i) - 40);
                        break;
                case 10:
                        j++;
                        background.SetTopLeft((MW*j), (MH*i));
                        background.ShowBitmap();
                        idoor.OnShow((MW*j) - 40, (MH*i) - 40);
                        break;
                case 11:
                        background.SetTopLeft((MW*j), (MH*i));
                        background.ShowBitmap();
                        break;
                case 12:
                        background.SetTopLeft((MW*j), (MH*i));
                        background.ShowBitmap();
                        slade.SetTopLeft((MW*j), (MH*i));
                        slade.ShowBitmap();
                        break;
                case 13:
                        background.SetTopLeft((MW*j), (MH*i));
                        background.ShowBitmap();
                        Rslade.SetTopLeft((MW*j), (MH*i));
                        Rslade.ShowBitmap();
                        break;
                case 14:
                        background.SetTopLeft((MW*j), (MH*i));
                        background.ShowBitmap();
                        break;
                case 15:
                        background.SetTopLeft((MW*j), (MH*i));
                        background.ShowBitmap();
                        Redbutton[Rbnum]->SetTopLeft((MW*j), (MH*i));
                        Redbutton[Rbnum++]->ShowBitmap();
                        break;
                case 16:
                        down.SetTopLeft((MW*j), (MH*i));
                        down.ShowBitmap();
                        break;
                case 17:
                        background.SetTopLeft((MW*j), (MH*i));
                        background.ShowBitmap();
                        break;
                case 18:
                        background.SetTopLeft((MW*j), (MH*i));
                        background.ShowBitmap();
                        break;
                case 20:
                        background.SetTopLeft((MW*j), (MH*i));
                        background.ShowBitmap();
                        Lever[Lnum++]->SetXY((MW*j), (MH*i));
                        break;
                case 21:
                        down.SetTopLeft((MW*j), (MH*i));
                        down.ShowBitmap();
                        break;
                default:
                        ASSERT(0);
                }
        }
        ////////////////////將物件建立在地圖上////////////////////
for (int i = 0; i < stonenum;i++) stone[i]->ShowBitmap();
for (int i = 0; i < elevatornum; i++) _elevator[i]->ShowBitmap();
for (int i = 0; i < fallwallnum; i++) fallwall[i]->ShowBitmap();
for (int i = 0; i < risingnum; i++) rising[i]->OnShow();
for (int i = 0; i < chainelevatornum; i++) chainelevator[i]->onShow();
for (int i = 0; i < Levernum; i++)
{
        Lever[i]->OnShow();
```

```
                LeverOn(Lever[i]->Getimage().IsFinalBitmap());
        }
        ////////////////////////陷阱流動//////////////////////////
        _fire.OnMove();
        _water.OnMove();
        _poison.OnMove();
        ////////////////////////////////////////////////////////
}
int CgameMap::floor(man *player,int move)
{
        ////////////////////抓取角色位置//////////////////////////////
        int i = ((player->GetX2() + player->GetX1()) / 2) / 20;
        int j = (player->GetY2()) / 20;
        ////////////////////判斷是否在石頭上//////////////////////////
        if (stone.size() != 0) if (!stone[0]->CheckStone(player->GetX1() + 5,player->GetX2()-5,player->GetY1()+5,player->GetY2()+5))    return
        stone[0]->GetY()-1;
        ////////////////////判斷腳下位置//////////////////////////
        switch (map[j][i]){
        case 0:
                switch (map[j-1][i]) {
                case 3:
                        player->SetXY(player->GetX1(), player->GetY1()+move);
                        return (_elevator[0]->GetY());
                case 4:
                        return (MH*j + 12);
                case 5:
                        return (MH*j + 12);
                case 6:
                        return (MH*j + 12);
                case 12:
                        return (OnSlade(player, i, j - 1));
                case 13:
                        return (OnRSlade(player, i, j - 1));
                case 14:
                for (i = 0; i < (int)fallwall.size(); i++)
                {
                        if (fallwall[i]->Onstand(*player))
                        {
                                player->SetXY(player->GetX1(), fallwall[i]->GetY()-39);
                                return (fallwall[i]->GetY());
                        }
                }
                return (MH*j);
                case 18:
                for (i = 0; i < (int)chainelevator.size(); i++)
                {
                        if (chainelevator[i]->Onstand(*player))
                        {
                                player->SetXY(player->GetX1(), chainelevator[i]->GetY() - 39);
                                return (chainelevator[i]->GetY());
                        }
                        else if (chainelevator[i]->Onstand2(*player))
                        {
                                player->SetXY(player->GetX1(), chainelevator[i]->GetY2()+ (int)chainelevator[i]->Getcount() - 39 +4);
                                return (player->GetY2());
                        }
                }
                case 20:
                        if (!Lever[0]->Getimage().IsFinalBitmap()) return (OnSlade(player, i, j-1));
                        else
                        {
                                player->setleft(1);
                                return (OnRSlade(player, i, j-1));
                        }
                default:
                        return 480;
                }
        case 1:
                switch (map[j - 1][i])
                {
                case 12:
                        return (OnSlade(player,i,j-1));
                case 13:
                        return (OnRSlade(player, i, j - 1));
                case 20:
                        if (!Lever[0]->Getimage().IsFinalBitmap()) return (OnSlade(player, i, j-1));
```

```
                        else
                        {
                                player->setleft(1);
                                return (OnRSlade(player, i, j-1));
                        }
                default:
                        break;
                }
                return (MH*j);
                break;
        case 2:
                return (MH*(j+1));
        case 3:
                player->SetXY(player->GetX1(), player->GetY1() + move);
                return (_elevator[0]->GetY());
        case 4:
                return (MH*j + 12);
        case 5:
                return (MH*j + 12);
        case 6:
                return (MH*j + 12);
        case 11:
                return (MH*j+3);
        case 12:
                return (OnSlade(player,i,j));
        case 13:
                player->setleft(1);
                return (OnRSlade(player, i, j));
        case 14:
                for (i = 0; i < (int)fallwall.size(); i++)
                {
                        if (fallwall[i]->Onstand(*player))
                        {
                                player->SetXY(player->GetX1(), fallwall[i]->GetY()-39);
                                return (fallwall[i]->GetY());
                        }
                }
                return (MH*j);
        case 15:
                return (MH*(j + 1));
        case 16:
                return (MH*j);
        case 17:
                return (MH*j);
        case 18:
                for (i = 0; i < (int)chainelevator.size(); i++)
                {
                        if (chainelevator[i]->Onstand(*player))
                        {
                                player->SetXY(player->GetX1(), chainelevator[i]->GetY() - 39);
                                return (chainelevator[i]->GetY());
                        }
                        else if (chainelevator[i]->Onstand2(*player))
                        {
                                player->SetXY(player->GetX1(), chainelevator[i]->GetY2() + (int)chainelevator[i]->Getcount() - 39);
                                return (player->GetY2());
                        }
                        return 480;
                }
        case 20:
                if (!Lever[0]->Getimage().IsFinalBitmap()) return (OnSlade(player, i, j));
                else
                {
                        player->setleft(1);
                        return (OnRSlade(player, i, j));
                }
        case 21:
                return (MH*j);
        default:
                return 480;
        }
}
int CgameMap::floor(man * player)
{
        ////////////////////////////////////////角色位置////////////////////////////////
        int i = ((player->GetX2() + player->GetX1()) / 2) / 20;
```

```
int j = (player->GetY2()) / 20;
//////////////////////////////////////////判斷是否在石頭上//////////////////////////
if (stone.size() != 0) if (!stone[0]->CheckStone(player->GetX1() + 5, player->GetX2() - 5, player->GetY1() + 5, player->GetY2() + 5))    return
stone[0]->GetY() - 1;
//////////////////////////////////////////判斷四周位置//////////////////////////////////
switch (map[j][i]) {
case 0:
        switch (map[j - 1][i]) {
        case 3:
                player->SetXY(player->GetX1(), (_elevator[0]->GetY()-39));
                return (_elevator[0]->GetY());
        case 4:
                return (MH*j + 12);
        case 5:
                return (MH*j + 12);
        case 6:
                return (MH*j + 12);
        case 12:
                return (OnSlade(player, i, j - 1));
        case 13:
                return (OnRSlade(player, i, j - 1));
        case 14:
                for (i = 0; i < (int)fallwall.size(); i++)
                {
                        if (fallwall[i]->Onstand(*player))
                        {
                                player->SetXY(player->GetX1(), fallwall[i]->GetY() - 39);
                                return (fallwall[i]->GetY());
                        }
                }
                return (MH*j);
        case 18:
                for (i = 0; i < (int)chainelevator.size(); i++)
                {
                        if (chainelevator[i]->Onstand(*player))
                        {
                                player->SetXY(player->GetX1(), chainelevator[i]->GetY() - 39);
                                return (chainelevator[i]->GetY());
                        }
                        else if (chainelevator[i]->Onstand2(*player))
                        {
                                player->SetXY(player->GetX1(), chainelevator[i]->GetY2() + (int)chainelevator[i]->Getcount() - 39 + 4);
                                return (player->GetY2());
                        }
                }
        case 20:
                if (!Lever[0]->Getimage().IsFinalBitmap()) return (OnSlade(player, i, j - 1));
                else
                {
                        player->setleft(1);
                        return (OnRSlade(player, i, j - 1));
                }
        default:
                return 480;
        }
case 1:
        switch (map[j - 1][i])
        {
        case 12:
                return (OnSlade(player, i, j - 1));
        case 13:
                return (OnRSlade(player, i, j - 1));
        case 20:
                if (!Lever[0]->Getimage().IsFinalBitmap()) return (OnSlade(player, i, j - 1));
                else
                {
                        player->setleft(1);
                        return (OnRSlade(player, i, j - 1));
                }
        default:
                break;
        }
        return (MH*j);
        break;
case 2:
        return (MH*(j + 1));
```

```
		case 3:
			player->SetXY(player->GetX1(), (_elevator[0]->GetY()-39));
			return (_elevator[0]->GetY());
		case 4:
			return (MH*j + 12);
		case 5:
			return (MH*j + 12);
		case 6:
			return (MH*j + 12);
		case 11:
			return (MH*j + 3);
		case 12:
			return (OnSlade(player, i, j));
		case 13:
			player->setleft(1);
			return (OnRSlade(player, i, j));
		case 14:
			for (i = 0; i < (int)fallwall.size(); i++)
			{
				if (fallwall[i]->Onstand(*player))
				{
					player->SetXY(player->GetX1(), fallwall[i]->GetY() - 39);
					return (fallwall[i]->GetY());
				}
			}
			return (MH*j);
		case 15:
			return (MH*(j + 1));
		case 16:
			return (MH*j);
		case 17:
			return (MH*j);
		case 18:
			for (i = 0; i < (int)chainelevator.size(); i++)
			{
				if (chainelevator[i]->Onstand(*player))
				{
					player->SetXY(player->GetX1(), chainelevator[i]->GetY() - 39);
					return (chainelevator[i]->GetY());
				}
				else if (chainelevator[i]->Onstand2(*player))
				{
					player->SetXY(player->GetX1(), chainelevator[i]->GetY2() + (int)chainelevator[i]->Getcount() - 39);
					return (player->GetY2());
				}
				return 480;
			}
		case 20:
			if (!Lever[0]->Getimage().IsFinalBitmap()) return (OnSlade(player, i, j));
			else
			{
				player->setleft(1);
				return (OnRSlade(player, i, j));
			}
		case 21:
			return (MH*j);
		default:
			return 480;
		}
		////////////////////////////////////////////////////////////////////////////
}
bool CgameMap::wall(man player)
{
		//////////////////////////////////抓取角色左上右上座標//////////////////////////
		int i1 = (player.GetX1() + 5) / 20;
		int j1 = (player.GetY1() + 5 - player.Getvelocity()) / 20;
		int i2 = (player.GetX2() - 5) / 20;
		int j2 = (player.GetY1() + 5 - player.Getvelocity()) / 20;
		//////////////////////////////////判斷頭上方塊//////////////////////////////////
		switch (map[j1][i1])
		{
		case 0:
			switch (map[j2][i2])
			{
			case 0:
				if (stone.size() != 0) return stone[0]->CheckStone(player.GetX1() + 5, player.GetX2() - 5, player.GetY1() + 5, player.GetY2()-3);
```

```
                else return true;
            case 1:
                return false;
            case 2:
                return false;
                break;
            case 7:
                if (stone.size() != 0) return stone[0]->CheckStone(player.GetX1() + 5, player.GetX2() - 5, player.GetY1() + 5, player.GetY2()-3);
                else return true;
            case 8:
                if (stone.size() != 0) return stone[0]->CheckStone(player.GetX1() + 5, player.GetX2() - 5, player.GetY1() + 5, player.GetY2()-3);
                else return true;
            default:
                return false;
            }
        case 1:
            return false;
        case 2:
            return false;
        case 7:
            switch (map[j2][i2])
            {
            case 0:
                if (stone.size() != 0) return stone[0]->CheckStone(player.GetX1() + 5, player.GetX2() - 5, player.GetY1() + 5, player.GetY2()-3);
                else return true;
            case 1:
                return false;
            case 2:
                return false;
            case 7:
                if (stone.size() != 0) return stone[0]->CheckStone(player.GetX1()+5, player.GetX2()-5, player.GetY1() + 5, player.GetY2()-3);
                else return true;
            case 8:
                if (stone.size() != 0) return stone[0]->CheckStone(player.GetX1()+5, player.GetX2()-5, player.GetY1() + 5, player.GetY2()-3);
                else return true;
            default:
                return false;
            }
        case 8:
            switch (map[j2][i2])
            {
            case 0:
                return true;
            case 1:
                return false;
            case 2:
                return false;
            case 7:
                if (stone.size() != 0) return stone[0]->CheckStone(player.GetX1()+5, player.GetX2()-5, player.GetY1() + 5, player.GetY2()-3);
                else return true;
            case 8:
                if (stone.size() != 0) return stone[0]->CheckStone(player.GetX1()+5, player.GetX2()-5, player.GetY1() + 5, player.GetY2()-3);
                else return true;
            default:
                return false;
            }
        case 15:
            return false;
        case 17:
            return false;
        case 18:
            return false;
        default:
            return false;
        }
        ////////////////////////////////////////////////////////////////////////////////////////////
}
int CgameMap::leftwall(man player)
{
        ////////////////////////////////////抓取角色左上左下位置//////////////////////////////////////
        int i1 = (player.GetX1() + 5 - player.Getstep()) / 20;
        int j1 = (player.GetY1() + 5) / 20;
        int j2 = (player.GetY2() - 6) / 20;
        ////////////////////////////////////判斷是否為石頭//////////////////////////////////////////////
        if(stone.size() != 0){
            if (!stone[0]->CheckStone(player.GetX1()-player.Getstep()+5, player.GetX1()      - player.Getstep()+5, player.GetY1()+5,
```

```
            player.GetY2()-3))
            {
                if (player.Onfloor())
                    PushStone(1);//////////推動石頭
                return 0;
            }
    }
    ////////////////////////////////////判斷左邊方塊/////////////////////////////////////////////////
    switch (map[j1][i1])
    {
    case 0:
            switch (map[j2][i1])
            {
            case 0:
                return 1;
            case 1:
                return 0;
            case 2:
                return 1;
            case 12:
                return 0;
            case 13:
                return 1;
            case 15:
                return 1;
            case 16:
                return 0;
            case 17:
                return 0;
            case 20:
                if (!Lever[0]->Getimage().IsFinalBitmap() && player.GetMovingLeft()) Lever[0]->OnMove();
                else return 1;
                return 0;
            case 21:
                return 0;
            default:
                return 1;
            }
            break;
    case 1:
            return 0;
            break;
    case 2:
            switch (map[j2][i1])
            {
            case 0:
                return 1;
            case 1:
                return 0;
                break;
            case 2:
                return 1;
            case 15:
                return 1;
            case 16:
                return 0;
            case 17:
                return 0;
            case 21:
                return 0;
            default:
                return 1;
            }
    case 12:
            return 0;
    case 13:
            return 1;
    case 15:
            switch (map[j2][i1])
            {
            case 0:
                return 1;
                break;
            case 1:
                return 0;
                break;
```

```
                case 2:
                        return 1;
                        break;
                case 15:
                        return 1;
                case 16:
                        return 0;
                case 17:
                        return 0;
                case 21:
                        return 0;
                default:
                        return 1;
                        break;
                }
        case 16:
                return 0;
        case 17:
                return 0;
        case 18:
                return 0;
        case 20:
                if (!Lever[0]->Getimage().IsFinalBitmap()&& player.GetMovingLeft()) Lever[0]->OnMove();
                else return 1;
                return 0;
        case 21:
                return 0;
        default:
                return 1;
                break;
        }
        ////////////////////////////////////////////////////////////////////////////////////////////
}
int CgameMap::rightwall(man *player)
{
        ////////////////////////////////////////抓取角色右上右下座標//////////////////////////////////
        int i1 = (player->GetX2() - 5 + player->Getstep()) / 20;
        int j1 = (player->GetY1() + 5) / 20;
        int j2 = (player->GetY2() - 6) / 20;
        ////////////////////////////////////////判斷是否為石頭//////////////////////////////////////////
        if (stone.size() != 0){
                if (!stone[0]->CheckStone(player->GetX2() + player->Getstep(), player->GetX2() +player->Getstep(), player->GetY1() + 5, player-
                >GetY2() - 3))
                {
                        if(player->Onfloor())
                                PushStone(2); //////////////推動石頭
                        return 0;
                }
        }
        ////////////////////////////////////////判斷右邊方塊//////////////////////////////////////////////
        switch (map[j1][i1])
        {
        case 0:
                switch (map[j2][i1])
                {
                case 0:
                        return 1;
                        break;
                case 1:
                        return 0;
                        break;
                case 2:
                        return 1;
                case 12:
                        return 1;
                        break;
                case 13:
                        return 0;
                case 15:
                        return 1;
                case 16:
                        return 0;
                case 17:
                        return 0;
                case 20:
                        if (Lever[0]->Getimage().IsFinalBitmap() && player->GetMovingRight()) Lever[0]->OnMove();
```

```
                        else return 1;
                        return 0;
                case 21:
                        return 0;
                default:
                        return 1;
                        break;
                }
                break;
        case 1:
                return 0;
        case 2:
                switch (map[j2][i1])
                {
                case 0:
                        return 1;
                case 1:
                        return 0;
                case 2:
                        return 1;
                case 15:
                        return 1;
                case 16:
                        return 0;
                case 17:
                        return 0;
                case 21:
                        return 0;
                default:
                        return 1;
                }
                break;
        case 12:
                return 1;
                break;
        case 13:
                return 0;
        case 15:
                switch (map[j2][i1])
                {
                case 0:
                        return 1;
                case 1:
                        return 0;
                case 2:
                        return 1;
                case 15:
                        return 1;
                case 16:
                        return 0;
                case 17:
                        return 0;
                case 21:
                        return 0;
                default:
                        return 1;
                }
                break;
        case 16:
                return 0;
        case 17:
                return 0;
        case 18:
                return 0;
        case 20:
                if (Lever[0]->Getimage().IsFinalBitmap() && player->GetMovingRight()) Lever[0]->OnMove();
                else return 1;
                return 0;
        case 21:
                return 0;
        default:
                return 1;
                break;
        }
        ////////////////////////////////////////////////////////////////////////////
}
```

```
void CgameMap::PushStone(int direction)
{
	//////////////////若有石頭/////////////////////////////////////////////////
	if (stonenum != 0)
	{
		////////////////////////////石頭往左推/////////////////////////////
		if (direction == 1)
		{
			if (map[stone[0]->GetY() / 20][((stone[0]->GetX() - 7) / 20)] != 1 || map[stone[0]->GetY() / 20 + 1][((stone[0]->GetX() - 7) /
			20)] != 1)
			{
				//////////////////////更新地圖/////////////////////////////
				map[stone[0]->GetY() / 20][stone[0]->GetX() / 20] = 0;
				map[stone[0]->GetY() / 20][(stone[0]->GetX() - 7) / 20] = 11;
				stone[0]->SetXY(stone[0]->GetX() - 7, stone[0]->GetY());
				stone[0]->ShowBitmap();
				//////////////////////////////////////////////////////////////
			}

		}
		////////////////////////////石頭往右推/////////////////////////////////
		else if (direction == 2)
		{
			if (map[stone[0]->GetY() / 20][((stone[0]->GetX() + 7) / 20) + 2] != 1 && map[stone[0]->GetY() / 20 + 1][((stone[0]->GetX() +
			7) / 20) + 2] != 1)
			{
				//////////////////////////////更新地圖/////////////////////////////////
				map[stone[0]->GetY() / 20][stone[0]->GetX() / 20] = 0;
				map[stone[0]->GetY() / 20][(stone[0]->GetX() + 7) / 20] = 11;
				stone[0]->SetXY(stone[0]->GetX() + 7, stone[0]->GetY());
				stone[0]->ShowBitmap();
				//////////////////////////////////////////////////////////////////
			}
		}
	}
}
void CgameMap::StoneDown()
{
	//////////////////////////////////////判斷是否有石頭////////////////////////////////
	if (stonenum != 0)
	{
		////////////////////////////////判斷石頭底下方塊/////////////////////////////////
		if ((map[(stone[0]->GetY() / 20) + 2][stone[0]->GetX() / 20] == 18 || map[(stone[0]->GetY() / 20) + 2][(stone[0]->GetX() / 20) + 1] ==
		18) || (map[(stone[0]->GetY() / 20) + 1][stone[0]->GetX() / 20] == 18 || map[(stone[0]->GetY() / 20) + 1][(stone[0]->GetX() / 20) + 1]
		== 18))
		{
			for (int i = 0; i < (int)chainelevator.size(); i++)
			{
				//////////////////////若石頭壓在電梯一上////////////////////////////
				if (chainelevator[i]->OnStand(stone[0]->GetX(), stone[0]->GetY()+40))
				{
					//////////////////////電梯下降地圖更新////////////////////////////
					map[stone[0]->GetY() / 20][stone[0]->GetX() / 20] = 0;
					stone[0]->SetXY(stone[0]->GetX(), chainelevator[i]->GetY() - 40);
					map[stone[0]->GetY() / 20][stone[0]->GetX() / 20] = 11;
					chainelevator[i]->SetFall(chainelevator[i]->GetFall()+1);
					//////////////////////////////////////////////////////////////////
				}
				//////////////////////若石頭壓在電梯二上////////////////////////////
				else if (chainelevator[i]->OnStand2(stone[0]->GetX(), stone[0]->GetY() + 40))
				{
					//////////////////////電梯下降地圖更新////////////////////////////
					map[stone[0]->GetY() / 20][stone[0]->GetX() / 20] = 0;
					stone[0]->SetXY(stone[0]->GetX(), chainelevator[i]->GetY2() + (int)chainelevator[i]->Getcount() - 40 );
					map[stone[0]->GetY() / 20][stone[0]->GetX() / 20] = 11;
					chainelevator[i]->SetFall2(chainelevator[i]->GetFall2()+1);
					//////////////////////////////////////////////////////////////////
				}
				//////////////////////////////////////////////////////////////////////
			}
		}
		/////////////////////////////////////若底下為空氣////////////////////////////
		else if (map[(stone[0]->GetY() / 20) + 2][stone[0]->GetX() / 20] == 0 && map[(stone[0]->GetY() / 20) + 2][(stone[0]->GetX() / 20) +
		1] == 0)
		{
			//////////////////////下降並更新地圖/////////////////////////////////
```

```
                map[stone[0]->GetY() / 20][stone[0]->GetX() / 20] = 0;
                map[stone[0]->GetY() / 20 + 1][stone[0]->GetX() / 20] = 11;
                stone[0]->SetXY(stone[0]->GetX(), stone[0]->GetY() + 5);
                stone[0]->ShowBitmap();
                ////////////////////////////////////////////////////////////////////////
            }
        }
    }
    int CgameMap::elevatormove(bool flag)
    {
        ////////////////////////////////電梯數大於 1////////////////////////////////
        if (elevatornum != 0)
        {
            if (flag)
            {
                ////////////////////////////電梯下降////////////////////////////////
                if (map[((_elevator[0]->GetY()-1) / 20)+1][_elevator[0]->GetX() / 20] != 1)
                {
                    int nowy = _elevator[0]->down();
                    for (int i = 0; i < 3; i++)
                    {
                        map[(_elevator[0]->GetY()- _elevator[0]->Getvelocity()) / 20-1][_elevator[0]->GetX() / 20 + i] = 0;
                        map[((_elevator[0]->GetY()-1) / 20)][_elevator[0]->GetX() / 20 + i] = 3;
                    }
                    _elevator[0]->SetTopLeft(_elevator[0]->GetX(), nowy);
                    _elevator[0]->ShowBitmap();
                    return _elevator[0]->Getvelocity();
                }
                ////////////////////////////////////////////////////////////////////////
            }
            else
            {
                ////////////////////////電梯上升////////////////////////////////////
                if (_elevator[0]->getoy() < _elevator[0]->GetY())
                {
                    int nowy = _elevator[0]->up();
                    for (int i = 0; i < 3; i++)
                    {
                        map[_elevator[0]->GetY() / 20][_elevator[0]->GetX() / 20 + i] = 3;
                        if(map[(_elevator[0]->GetY() / 20) + 1][_elevator[0]->GetX() / 20 + i]==3)
                        map[(_elevator[0]->GetY() / 20) + 1][_elevator[0]->GetX() / 20 + i] = 0;
                    }
                    _elevator[0]->SetTopLeft(_elevator[0]->GetX(), nowy);
                    _elevator[0]->ShowBitmap();
                    return _elevator[0]->Getvelocity()*-1;
                }
                ////////////////////////////////////////////////////////////////////////
            }
        }
        return 0;
    }

    void CgameMap::FallWallMove(int j)
    {
        ////////////////////////////若是角色站在上方////////////////////////////////
        if (fallwall[j]->GetFall()>0)
        {
            ////////////////////////////地板下陷////////////////////////////////////
            if (map[((fallwall[j]->GetY()) / 20) + 1][fallwall[j]->GetX() / 20] != 1)
            {
                fallwall[j]->Down();
                for (int i = 0; i < 3; i++)
                {
                    map[(fallwall[j]->GetY() - 4) / 20 - 1][fallwall[j]->GetX() / 20 + i] = 0;
                    map[((fallwall[j]->GetY()-4) / 20)][fallwall[j]->GetX() / 20 + i] = 14;
                }
                fallwall[j]->ShowBitmap();
            }
            ////////////////////////////////////////////////////////////////////////
        }
        else
        {
            ////////////////////////////地板上升回地面////////////////////////////////
            if (fallwall[j]->Getoy() < fallwall[j]->GetY())
            {
            fallwall[j]->Up();
```

```
                for (int i = 0; i < 3; i++)
                {
                        map[fallwall[j]->GetY() / 20][fallwall[j]->GetX() / 20 + i] = 14;
                        if (map[(fallwall[j]->GetY() / 20) + 1][fallwall[j]->GetX() / 20 + i] == 14) map[(fallwall[j]->GetY() / 20) + 1][fallwall[j]-
                        >GetX() / 20 + i] = 0;
                }
                fallwall[j]->ShowBitmap();
                }
                ////////////////////////////////////////////////////////////////////////////
        }
}

void CgameMap::ChainElevatorMove(int j)
{
        ////////////////////若電梯一重量大於電梯二//////////////////////////////////////
        if (chainelevator[j]->GetFall() >chainelevator[j]->GetFall2())
        {
                ////////////////電梯 1 下降、電梯 2 上升////////////////////////////
                if (chainelevator[j]->GetY()<chainelevator[j]->GetaimY1())
                {
                        chainelevator[j]->Down();
                        for (int i = 0; i < 3; i++)
                        {
                                map[(chainelevator[j]->GetY() - 4) / 20 - 1][chainelevator[j]->GetX() / 20 + i] = 0;
                                map[((chainelevator[j]->GetY() - 4) / 20)][chainelevator[j]->GetX() / 20 + i] = 18;
                                map[(chainelevator[j]->GetY2()+ (int)chainelevator[j]->Getcount()) / 20][chainelevator[j]->GetX2() / 20 + i] = 18;
                                if (map[((chainelevator[j]->GetY2() + (int)chainelevator[j]->Getcount()) / 20) + 1][chainelevator[j]->GetX2() / 20 + i] ==
                                18) map[((chainelevator[j]->GetY2() + (int)chainelevator[j]->Getcount()) / 20) + 1][chainelevator[j]->GetX2() / 20 + i] =
                                0;
                        }
                        chainelevator[j]->ShowBitmap();
                }
                ////////////////////////////////////////////////////////
        }
                ////////////////電梯 2 重量大於電梯 1////////////////////////////
        else if(chainelevator[j]->GetFall2() > chainelevator[j]->GetFall())
        {
                ////////////////電梯 1 上升、電梯 2 下降//////////////////////////////////////
                if ((int)chainelevator[j]->Getcount()<=0)
                {
                        chainelevator[j]->Up(4);
                        for (int i = 0; i < 3; i++)
                        {
                                map[chainelevator[j]->GetY() / 20][chainelevator[j]->GetX() / 20 + i] = 18;
                                if (map[(chainelevator[j]->GetY() / 20) + 1][chainelevator[j]->GetX() / 20 + i] == 18) map[(chainelevator[j]->GetY() /
                                20) + 1][chainelevator[j]->GetX() / 20 + i] = 0;
                                map[(chainelevator[j]->GetY2() + (int)chainelevator[j]->Getcount()) / 20 - 1][chainelevator[j]->GetX2() / 20 + i] = 0;
                                map[((chainelevator[j]->GetY2() + (int)chainelevator[j]->Getcount()) / 20)][chainelevator[j]->GetX2() / 20 + i] = 18;
                        }
                        chainelevator[j]->ShowBitmap();
                }
                ////////////////////////////////////////////////////////////////////////////
        }
}

bool CgameMap::gameover(man player)
{
        ////////////////////抓取角色位置//////////////////////////////////
        int i = ((player.GetX2() + player.GetX1()) / 2) / 20;
        int j = (player.GetY2()) / 20;
        //////////////若是進到與自身角色不同的陷阱或是毒素死亡/////////////////
        switch (map[j][i]) {
        case 4:
                if (player.Getcharacteristic() == "ice") return true;
                else return false;
                break;
        case 5:
                if (player.Getcharacteristic() == "fire") return true;
                else return false;
                break;
        case 6:
                return true;
                break;
        default:
                break;
                ASSERT(0);
```

```
		}
		switch (map[j-1][i]){
		case 4:
			if (player.Getcharacteristic() == "ice") return true;
			else return false;
			break;
		case 5:
			if (player.Getcharacteristic() == "fire") return true;
			else return false;
			break;
		case 6:
			return true;
			break;
		default:
			return false;
			ASSERT(0);

		}
		//////////////////////////////////////////////////////
}
bool CgameMap::isgem(man player)
{
		//////////////////抓取角色四個點//////////////////////
		int manY1 = player.GetY1();
		int manY2 = player.GetY2();
		int manX1 = player.GetX1();
		int manX2 = player.GetX2();
		int image[4][2] = { 0 };
		image[0][0] = manX1 + 5;
		image[0][1] = manY1 + 5;
		image[1][0] = manX2 - 5;
		image[1][1] = manY1 + 5;
		image[2][0] = manX1;
		image[2][1] = manY2 - 1;
		image[3][0] = manX2;
		image[3][1] = manY2 - 1;
		////////////判斷是否碰撞到適合的寶石//////////////////
		switch (map[image[0][1] / 20][image[0][0] / 20]) {
		case 7:
			if (player.Getcharacteristic() == "fire")
			{
				map[image[0][1] / 20][image[0][0] / 20] = 0;
				return true;
			}
			else break;
		case 8:
			if (player.Getcharacteristic() == "ice")
			{
				map[image[0][1] / 20][image[0][0] / 20] = 0;
				return true;
			}
			else break;
		default:
			break;
		}
		switch (map[image[1][1] / 20][image[1][0] / 20]) {
		case 7:
			if (player.Getcharacteristic() == "fire")
			{
				map[image[1][1] / 20][image[1][0] / 20] = 0;
				return true;
			}
			else break;
		case 8:
			if (player.Getcharacteristic() == "ice")
			{
				map[image[1][1] / 20][image[1][0] / 20] = 0;
				return true;
			}
			else break;
		default:
			break;
		}
		switch (map[image[2][1] / 20][image[2][0] / 20]) {
		case 7:
			if (player.Getcharacteristic() == "fire")
```

```
                {
                        map[image[2][1] / 20][image[2][0] / 20] = 0;
                        return true;
                }
                else break;
        case 8:
                if (player.Getcharacteristic() == "ice")
                {
                        map[image[2][1] / 20][image[2][0] / 20] = 0;
                        return true;
                }
                else break;
        default:
                break;
        }
        switch (map[image[3][1] / 20][image[3][0] / 20]) {
        case 7:
                if (player.Getcharacteristic() == "fire")
                {
                        map[image[3][1] / 20][image[3][0] / 20] = 0;
                        return true;
                }
                else return false;
        case 8:
                if (player.Getcharacteristic() == "ice")
                {
                        map[image[3][1] / 20][image[3][0] / 20] = 0;
                        return true;
                }
                else return false;
        default:
                return false;
                ASSERT(0);
        }
        ////////////////////////////////////////////////////
}

bool CgameMap::Clearance(man player)
{
        ////////////////////角色位置//////////////////////
        int i = ((player.GetX2() + player.GetX1()) / 2) / 20;
        int j = (player.GetY2() - 20) / 20;
        ////////////////判斷關卡是否完成//////////////////
        switch (map[j][i]) {
        case 9:
                return fdoor.Clearance(player, firegemnum);
                break;
        case 10:
                return idoor.Clearance(player, icegemnum);
        default:
                if (fdoor.GetDoor()->GetCurrentBitmapNumber()!= 0 && player.Getcharacteristic() == "fire")
                {
                        fdoor.GetDoor()->Movetonum(fdoor.GetDoor()->GetCurrentBitmapNumber() - 1);
                }
                if (idoor.GetDoor()->GetCurrentBitmapNumber() != 0 && player.Getcharacteristic() == "ice")
                {
                        idoor.GetDoor()->Movetonum(idoor.GetDoor()->GetCurrentBitmapNumber() - 1);
                }
                return false;
                ASSERT(0);
        }
        ////////////////////////////////////////////////////
}

int CgameMap::OnSlade(man* player, int i, int j)
{
        int temp = 0;
        ////////////////將角色位置根據移動的 X 座標上升 Y 座標增加//////////////////////////
        player->SetXY(player->GetX1() - 1, player->GetY1());
        temp = (player->GetX2()-(i+1)*20);
        if (player->GetX2() >= i * 20&& player->GetMovingRight()) player->SetXY(player->GetX1() + player->Getstep(), ((j - 1) * 20) - temp);
        player->SetXY(player->GetX1(), ((j-1) *20) - temp);
        return player->GetY2();
        ////////////////////////////////////////////////////////////////////////////
}
```

```
int CgameMap::OnRSlade(man * player, int i, int j)
{
	int temp = 0;
	//////////////將 X 座標上升 Y 下降////////////////////////////////////////////
	player->SetXY(player->GetX1() + 1, player->GetY1());
	temp = 20-(player->GetX1()-((i-1)*20));
	if (player->GetX1()<= (i) * 20 && player->GetMovingLeft())
	{
		player->SetXY(player->GetX1()-player->Getstep(), ((j - 1) * 20) -temp);
	}
	else player->SetXY(player->GetX1(), ((j - 1) * 20) - temp);
	return player->GetY2();
	//////////////////////////////////////////////////////////////////////////
}

void CgameMap::makewall(bool flag)
{
	int wallx=0, wally=0,count=0;
	////////////////////////////////找到牆的位子////////////////////////////////
	for (int i = 0; i < 24; i++) for (int j = 0; j < 32; j++) if (map[i][j] == 16) { wallx = j, wally = i; count++; }
	////////////////////////////////若是按鈕被觸發長出一個牆//////////////////
	if (!flag)
	{
		if (map[wally + 1][wallx] == 0 && delaycount-- <= 0)
		{
			delaycount = 5;
			map[wally + 1][wallx] = 16;
		}
	}
	//////////////////////////////若是沒被觸發牆縮回去////////////////////////////
	else
	{
		if (count > 1 && delaycount-- <= 0)
		{
			delaycount = 5;
			map[wally][wallx] = 0;
		}
	}
	//////////////////////////////////////////////////////////////////////////
}
void CgameMap::LeverOn(bool flag)
{
	int wallx = 0, wally = 0, count = 0;
	////////////////////////////////找到牆下降的位置//////////////////////////
	for (int i = 0; i < 24; i++) for (int j = 0; j < 32; j++) if (map[i][j] == 21) { if (count == 0) { wallx = j, wally = i; } count++;}
	////////////////////////////////若是拉桿被觸發//////////////////////////////
	if (!flag)
	{
	////////////////////////////////將牆壁下降//////////////////////////////////
		if (map[wally - 1][wallx] == 0 && delaycount-- <= 0)
		{
			delaycount = 5;
			map[wally - 1][wallx] = 21;
		}
	}
	else
	{
	////////////////////////////////牆壁上升//////////////////////////////////
		if (count > 1 && delaycount-- <= 0)
		{
			delaycount = 5;
			map[wally][wallx] = 0;
		}
	}
}
void CgameMap::deleteObject()
{
	////////////////////////////////刪除物件//////////////////////////////////
	for (int i = 0; i < buttonnum; i++) delete button[i];
	for (int i = 0; i < buttonnum; i++) button.erase(button.begin());
	for (int i = 0; i < elevatornum; i++) delete _elevator[i];
	for (int i = 0; i < elevatornum; i++) _elevator.erase(_elevator.begin());
	for (int i = 0; i < stonenum; i++) delete stone[i];
	for (int i = 0; i < stonenum; i++) stone.erase(stone.begin());
	for (int i = 0; i < fallwallnum; i++) delete fallwall[i];
	for (int i = 0; i < fallwallnum; i++) fallwall.erase(fallwall.begin());
```

```
            for (int i = 0; i < redbuttonnum; i++) delete Redbutton[i];
            for (int i = 0; i < redbuttonnum; i++) Redbutton.erase(Redbutton.begin());
            for (int i = 0; i < risingnum; i++) delete rising[i];
            for (int i = 0; i < risingnum; i++) rising.erase(rising.begin());
            for (int i = 0; i < chainelevatornum; i++) delete chainelevator[i];
            for (int i = 0; i < chainelevatornum; i++) chainelevator.erase(chainelevator.begin());
            for (int i = 0; i < Levernum; i++) delete Lever[i];
            for (int i = 0; i < Levernum; i++) Lever.erase(Lever.begin());
            ////////////////////////////////////////////////////////////////////////////
        }
}
```

man.h

```
#ifndef MAN_H
#define MAN_H
#include "bullet.h"
namespace game_framework {
    class man
    {
    public:
        man();
        int     GetX1();                    // 角色左上角 x 座標
        int     GetY1();                    // 角色左上角 y 座標
        int     GetX2();                    // 角色右下角 x 座標
        int     GetY2();                    // 角色右下角 y 座標
        int     Getvelocity();
        int     Getfloor();
        bool Onfloor();
        string Getcharacteristic();
        bool GetRising();
        bool Getfly();
        void Addgem();
        int getgem();
        int Getstep();
        virtual void Initialize();          // 設定角色為初始值
        virtual void LoadBitmap();          // 載入圖形
        virtual void LoadBitmap(int);
        void OnMove();                      // 移動角色
        void OnMove(man *player);
        void setfloor(int _floor);
        void sethead(bool flag);
        void setleft(int flag);
        void setright(int flag);
        void jump();                        // 跳躍
        void jump(int);
        void OnShow();                      // 將角色圖形貼到畫面
        void OnShow(int);
        void SetMovingDown(bool flag);    // 設定是否正在往下移動
        void SetMovingLeft(bool flag);    // 設定是否正在往左移動
        void SetMovingRight(bool flag); // 設定是否正在往右移動
        void SetMovingUp(bool flag);      // 設定是否正在往上移動
        void SetTwojump(bool flag);
        void SetAttack(bool flag);
        void SetAttacked(int flag);
        bool SetShooting();
        void SetRush();
        void SetisMagic(bool flag);
        CAnimation GetAnimation();
        bool GetMovingLeft();
        bool GetMovingRight();
        bool Getleft();
        bool Getright();
        bool Getshooting();
        bool GetRush();
        void SetXY(int nx, int ny);         // 設定角色左上角座標
        void Setfly(bool flag);
        int GetBullet();
        int GetDamage();
        void AddDamage();
        void SetCombodelay();
        virtual void jumpsound();
        void DeleteBullet();
    protected:
        int floor;
        int velocity;                   // 目前的速度(點/次)
        int initial_velocity;    // 初始速度
        bool rising;                    // true 表上升、false 表下降
```

```
		bool fly;
		int x, y;                                   // 角色左上角座標
		bool isMovingDown;                  // 是否正在往下移動
		bool isMovingLeft;                  // 是否正在往左移動
		bool isMovingRight;                 // 是否正在往右移動
		bool isMovingUp;                    // 是否正在往上移動
		bool isTwojump;
		bool isAttack;
		bool isRush;
		bool isMagic;
		int isAttacked;
		int RushTime;
		bool isShooting;
		bool head;
		int direction;
		int combodelay;
		int times;
		int left;
		int step_size;
		int magic;
		string characteristic;
		int right;
		int gem;
		int damage;
		CAnimation animation;               // 利用動畫作圖形
		CMovingBitmap MagicBar;
		CMovingBitmap Black;
		int MagicX, MagicY;
		vector <bullet*> shoot;
		vector <char> combo;
	};
}
#endif
```

man.cpp

```
#include "stdafx.h"
#include "Resource.h"
#include <mmsystem.h>
#include <ddraw.h>
#include "audio.h"
#include "gamelib.h"
#include "man.h"

namespace game_framework {
	man::man()
	{
		Initialize();
	}
	int man::GetX1()
	{
		return x;
	}

	int man::GetY1()
	{
		return y;
	}

	int man::GetX2()
	{
		return x + 31 ;
	}

	int man::GetY2()
	{
		return y + 39;
	}

	int man::Getvelocity()
	{
		return velocity;
	}
	int man::Getfloor()
	{
		return floor;
	}
	bool man::Onfloor()
```

```
{
	if (animation.GetCurrentBitmapNumber() < 14 ||animation.GetCurrentBitmapNumber()>22) return true;
	else return false;
}
string man::Getcharacteristic()
{
	return characteristic;
}
bool man::GetRising()
{
	return rising;
}
bool man::Getfly()
{
	return fly;
}
void man::Addgem()
{
	gem++;
}
int man::getgem()
{
	return gem;
}
int man::Getstep()
{
	return step_size;
}

void man::Initialize()
{
}
void man::LoadBitmap()
{
}
void man::LoadBitmap(int i)
{
}
void man::OnMove()
{
	animation.SetDelayCount(1);
	//////////////////////////////////////////////////////判斷是否移動//////////////////////////////////////////////////
	if (isMovingLeft || isMovingRight || isMovingUp)
	{
		//////////////////////////////////////////////////////左移//////////////////////////////////////////////////
		if (isMovingLeft)
		{
			if (left == 1)
				x -= step_size;
			if (!(animation.GetCurrentBitmapNumber() > 7 && animation.GetCurrentBitmapNumber() < 14))
			{
				animation.Movetonum(8);
			}
			else
			{
				animation.OnMove();
			}
		}
		//////////////////////////////////////////////////////右移//////////////////////////////////////////////////
		else if (isMovingRight)
		{
			if (right == 1)
				x += step_size;
			if (!(animation.GetCurrentBitmapNumber() > 0 && animation.GetCurrentBitmapNumber() < 7))
			{
				animation.Movetonum(1);
			}
			else
			{
				animation.OnMove();
			}
		}
		//////////////////////////////////////////////////////跳躍//////////////////////////////////////////////////
		if ((isMovingUp && rising == false && velocity == initial_velocity && y >= floor - 39))
		{
			rising = true;
```

```
                velocity = initial_velocity;
                if (!(animation.GetCurrentBitmapNumber() > 14 && animation.GetCurrentBitmapNumber() < 18))
                {
                        animation.Movetonum(15);
                }
        }
    }
    ////////////////////////////////////////////////////站在原地//////////////////////////////////
    else
        animation.Reset();
    jump();
}

void man::OnMove(man *player)
{
    //--------子彈---------
    if (isShooting && isAttacked == 2)
    {
        shoot.push_back(new bullet(this->GetX1(), this->GetY1(),direction,characteristic));
        isShooting = false;
    }
    if (shoot.size() != 0) {
        for (int count = 0; count < this->GetBullet(); count++) {
            shoot[count]->Move();
            if (shoot[count]->Getbullet().GetCurrentBitmapNumber()==12 || shoot[count]->Getbullet().GetCurrentBitmapNumber() == 23 ||
            shoot[count]->Getbullet().GetCurrentBitmapNumber() == 35 || shoot[count]->Getbullet().GetCurrentBitmapNumber() == 47) {
                delete shoot[count];
                shoot.erase(shoot.begin()+count);
            }
            else if (((shoot[count]->GetX2()>player->GetX1()&&shoot[count]->Getdirection()==1 && shoot[count]->GetX1() < player-
            >GetX2()) ||    (shoot[count]->GetX1() < player->GetX2()&&shoot[count]->Getdirection() == 0&& shoot[count]->GetX2() >
            player->GetX1()))&&((player->GetY1()<shoot[count]->GetY2() && player->GetY1()>shoot[count]->GetY1())||(player-
            >GetY2()>shoot[count]->GetY1()&&player->GetY2()<shoot[count]->GetY2()))) {
                player->SetAttacked(shoot[count]->Getdirection());
                delete shoot[count];
                shoot.erase(shoot.begin() + count);
            }
        }
    }
    ////////////////////////////////////////////////////////////////////////////////////////////////
    animation.SetDelayCount(1);
    ////////////////////判斷使用衝鋒技能/////////////////////
    if (isRush)
    {
        ////////////////////////////若是衝鋒還有時間/////////////////////////
        if (direction == 0 && RushTime>0)
        {
            ////////////////////////////////向左衝刺/////////////////////////
            if (left == 1)
                x -= step_size*2;
            if (!(animation.GetCurrentBitmapNumber() > 7 && animation.GetCurrentBitmapNumber() < 13))
            {
                animation.Movetonum(8);
            }
            else
            {
                animation.OnMove();
            }
            direction = 0;
            ////////////////////////////////////////////////////////////
        }
        else if (direction == 1 && RushTime > 0)
        {
            ////////////////////////////////向右衝刺/////////////////////////
            if (right == 1)
                x += step_size * 2;
            if (!(animation.GetCurrentBitmapNumber() > 1 && animation.GetCurrentBitmapNumber() < 7))
            {
                animation.Movetonum(2);
            }
            else
            {
                animation.OnMove();
            }
            direction = 1;
            ////////////////////////////////////////////////////////////
```

```
        }
        if (RushTime-- <= 0)
            isRush = false;//結束衝刺
    }
    ////////////////////集氣//////////////////////
    else if (isMagic && isAttacked == 2)
    {
        animation.SetDelayCount(5);
        if(magic<100) magic++;//增加魔力
        ////////////////////////////切換圖片/////////////////////////////////////////////
        if (direction == 0)
        {
            if(animation.GetCurrentBitmapNumber()!=40) animation.Movetonum(40);
            else animation.Movetonum(41);
        }
        else if (direction == 1)
        {
            if (animation.GetCurrentBitmapNumber() != 38) animation.Movetonum(38);
            else animation.Movetonum(39);
        }
        ////////////////////////////////////////////////////////////////////////////
    }
    ////////////////////防禦//////////////////////
    else if (isMovingDown && isAttacked == 2)
    {
        if (direction == 0) animation.Movetonum(37);
        else if (direction == 1) animation.Movetonum(36);
    }
    ////////////////////移動//////////////////////
    else if ((isMovingLeft || isMovingRight || isMovingUp) && isAttacked ==2)
    {
        ////////////////////////////////////左移/////////////////////////////////////
        if (isMovingLeft)
        {
            if (left == 1)
                x -= step_size;
            if (!(animation.GetCurrentBitmapNumber() > 7 && animation.GetCurrentBitmapNumber() < 13))
            {
                animation.Movetonum(8);
            }
            else
            {
                animation.OnMove();
            }
            direction = 0;
        }
        ////////////////////////////////////////////////////右移/////////////////////////////////////////////
        else if (isMovingRight)
        {
            if (right == 1)
                x += step_size;
            if (!(animation.GetCurrentBitmapNumber() > 1 && animation.GetCurrentBitmapNumber() < 7))
            {
                animation.Movetonum(2);
            }
            else
            {
                animation.OnMove();
            }
            direction = 1;
        }
        ////////////////////////////////////////////////////跳躍/////////////////////////////////////////////
        if ((isMovingUp && rising == false && velocity == initial_velocity && y >= floor - 39) || (isMovingUp && isTwojump))
        {
            if (!(isMovingUp && rising == false && velocity == initial_velocity && y >= floor - 39))
                isTwojump = false;
            rising = true;
            isMovingUp = false;
            velocity = initial_velocity;

        }
        ////////////////////////////////////////////////////////////////////////////////////////////////
    }
    else if (isAttack && isAttacked==2)
    {
    ////////////////////////////////////////////////攻擊命中且對手未防禦或破防/////////////////////////////////////////
```

```
		if (x + 59 > player->GetX1() && player->GetX1() > x && direction ==1 && player->GetY1() < y + 10 && player->GetY1() > y - 10
		&&(player->GetAnimation().GetCurrentBitmapNumber()!=37 || player->GetDamage() > 5))
		{
			if (!(animation.GetCurrentBitmapNumber() > 27 && animation.GetCurrentBitmapNumber() <= 31)) animation.Movetonum(28);
			else if (animation.GetCurrentBitmapNumber() == 31)
			{
				player->SetAttacked(1);//將對手設定為被擊中
				isAttack = false;
				CAudio::Instance()->Play(9, false);
			}
			else animation.OnMove();

		}
		else if (x-20 < player->GetX2() && player->GetX2() < GetX2() && direction == 0 && player->GetY1() < y + 10 && player->GetY1()
		> y - 10 && direction == 0 && (player->GetAnimation().GetCurrentBitmapNumber() != 36 || player->GetDamage()>5))
		{
			if (!(animation.GetCurrentBitmapNumber() > 31 && animation.GetCurrentBitmapNumber() <= 35)) animation.Movetonum(32);
			else if (animation.GetCurrentBitmapNumber() == 35)
			{
				player->SetAttacked(0);
				isAttack = false;
				CAudio::Instance()->Play(9, false);
			}
			else animation.OnMove();
		}
		//////////////////////////////////////////////////攻擊未命中//////////////////////////////////////////////////
		else if (direction == 1)
		{
			if (animation.GetCurrentBitmapNumber() < 28)animation.Movetonum(28);
			else if (animation.GetCurrentBitmapNumber() == 31) isAttack = false;
			else animation.OnMove();
		}
		else if (direction == 0)
		{
			if (animation.GetCurrentBitmapNumber() < 32)animation.Movetonum(32);
			else if (animation.GetCurrentBitmapNumber() == 35) isAttack = false;
			else animation.OnMove();
		}
		//////////////////////////////////////////////////擊中防禦狀態增加破防值//////////////////////////////////////////
			if (((x + 59 > player->GetX1() && player->GetX1() > x && direction == 1 && player->GetY1() < y + 10 && player->GetY1()
			> y - 10 )||(x - 20 < player->GetX2() && player->GetX2() < GetX2() && player->GetY1() < y + 10 && player->GetY1() > y -
			10)) && player->GetAnimation().GetCurrentBitmapNumber() - 36 == direction &&(animation.GetCurrentBitmapNumber() ==
			31|| animation.GetCurrentBitmapNumber() == 35))
			player->AddDamage();
	}
	//////////////////////////////////////////被擊中//////////////////////////////////////////////////////////
	else if (isAttacked!=2)
	{
	//////////////////////////////向後移動/////////////////////////////////
		if (isAttacked == 0 && left == 1)
			x -= step_size/2;
		else if (isAttacked == 1 && right == 1)
			x += step_size/2;
		if (times == 24)
		{
			times = 0;
			isAttacked = 2;
		}
		else
		{
			if (times % 4 == 0 && player->GetX1() >= x && left == 1)
			{
				if (animation.GetCurrentBitmapNumber() >= 16 && animation.GetCurrentBitmapNumber() < 21)
					animation.OnMove();
				else
					animation.Movetonum(16);
			}
			else if (times % 4 == 0 && player->GetX1() < x && left == 1)
			{
				if (animation.GetCurrentBitmapNumber() >= 21 && animation.GetCurrentBitmapNumber() < 27)
					animation.OnMove();
				else
					animation.Movetonum(21);
			}
			times++;
		}
```

```
		//////////////////////////////////////////////////////////////////
	}
	else
	////////////////////////////////站在原地//////////////////////////////////////////
		if(direction ==1 ) animation.Reset();
		else if(direction == 0) animation.Movetonum(1);
	//////////////////////////////////////////////////////////////////
	jump(1);
}

void man::setfloor(int _floor)
{
	floor = _floor;
}

void man::sethead(bool flag)
{
	head = flag;
}

void man::setleft(int flag)
{
	left = flag;
}

void man::setright(int flag)
{
	right = flag;
}

void man::jump()
{
	if (rising)				// 上升狀態
	{
		if(velocity == initial_velocity)	jumpsound();
		if (velocity > 0)
		{
			if (head)
				y -= velocity;  // 當速度 > 0 時，y 軸上升(移動 velocity 個點，velocity 的單位為 點/次)
			if (fly && velocity < initial_velocity) velocity++;
			else velocity--;		// 受重力影響，下次的上升速度降低
		}
		else
		{
			rising = false; // 當速度 <= 0，上升終止，下次改為下降
			velocity = 1;   // 下降的初速(velocity)為 1
		}
		if (animation.GetCurrentBitmapNumber() > 14 && animation.GetCurrentBitmapNumber() < 18)
			animation.OnMove();
		else
		{
			animation.Movetonum(15);
		}
	}
	else				// 下降狀態
	{
		if (y < floor - 39)		// 當 y 座標還沒碰到地板
		{
			y += velocity; // y 軸下降(移動 velocity 個點，velocity 的單位為 點/次)
			if (fly) velocity-=2;
			else velocity++;			// 受重力影響，下次的下降速度增加

			if (animation.GetCurrentBitmapNumber() > 18 && animation.GetCurrentBitmapNumber() < 22)
				animation.OnMove();
			else
				animation.Movetonum(19);
		}
		else
		{
			y = floor - 39;   // 當 y 座標低於地板，更正為地板上
			velocity = initial_velocity; //  重設上升初始速度
		}
	}
}
void man::jump(int i)
{
```

```
        if (rising)                             // 上升狀態
        {
                if (velocity == initial_velocity)       jumpsound();
                if (velocity > 0)
                {
                        if (head)
                                y -= velocity;  // 當速度 > 0 時，y 軸上升(移動 velocity 個點，velocity 的單位為 點/次)
                        if (fly && velocity < initial_velocity) velocity++;
                        else velocity--;        // 受重力影響，下次的上升速度降低
                }
                else
                {
                        rising = false; // 當速度 <= 0，上升終止，下次改為下降
                        velocity = 1;   // 下降的初速(velocity)為 1
                }
                if (direction == 1)
                        animation.Movetonum(14);
                if (direction == 0)
                        animation.Movetonum(15);
        }
        else                            // 下降狀態
        {
                if (y < floor - 39)         // 當 y 座標還沒碰到地板
                {
                        y += velocity; // y 軸下降(移動 velocity 個點，velocity 的單位為 點/次)
                        if (fly) velocity -= 2;
                        else velocity++;                // 受重力影響，下次的下降速度增加 0
                        if (direction == 1)
                                animation.Movetonum(14);
                        if (direction == 0)
                                animation.Movetonum(15);
                }
                else
                {
                        y = floor - 39;   // 當 y 座標低於地板，更正為地板上
                        velocity = initial_velocity; // 重設上升初始速度
                }

        }
}
void man::SetMovingDown(bool flag)
{
        isMovingDown = flag;
        if (flag) combo.push_back('D');//將 COMBO 指令丟進 COMBO 表
        if (flag) damage = 0;
}

void man::SetMovingLeft(bool flag)
{
        isMovingLeft = flag;
        if (flag) combo.push_back('L');//將 COMBO 指令丟進 COMBO 表
}

void man::SetMovingRight(bool flag)
{
        isMovingRight = flag;
        if (flag) combo.push_back('R');//將 COMBO 指令丟進 COMBO 表
}

void man::SetMovingUp(bool flag)
{
        isMovingUp = flag;
        if(flag) combo.push_back('U');//將 COMBO 指令丟進 COMBO 表
}
bool man::SetShooting()
{
        if((int)combo.size() ==3 && combo[2]=='A' && (combo[1] == 'R' || combo[1] == 'L') && combo[0] == 'D' &&magic>=20)//若是按下防前攻
        的指令
        {
                isShooting = true;//射出子彈
                magic -= 20;
                for (int i = 0; i < (int)combo.size(); i++) combo.erase(combo.begin());//清除 COMBO 表
                return true;
        }
        return false;
}
```

```
void man::SetRush()
{
	if ((int)combo.size() > 3) combo.erase(combo.begin());//清除 COMBO 表
	if ((int)combo.size() == 3 && combo[2] == 'A' && ((combo[1] == 'R' && combo[0] == 'R') || (combo[1] == 'L'&& combo[0] ==
	'L'))&&magic>=20)//若是按下前前攻的指令
	{
		isRush = true;//向前衝刺
		RushTime = 10;
		for (int i = 0; i < (int)combo.size(); i++) combo.erase(combo.begin());//清除 COMBO 表
		magic -= 20;
	}
	if (combodelay-- <= 0)
	{
		for (int i = 0; i < (int)combo.size(); i++) combo.erase(combo.begin());//清除 COMBO 表
	}

}

void man::SetisMagic(bool flag)
{
	isMagic = flag;
}

CAnimation man::GetAnimation()
{
	return animation;
}

void man::SetTwojump(bool flag)
{
	isTwojump = flag;
}

void man::SetAttack(bool flag)
{
	isAttack = flag;
	if (flag) combo.push_back('A');//將 COMBO 指令丟進 COMBO 表
}

void man::SetAttacked(int flag)
{
	isAttacked = flag;
}
bool man::Getshooting()
{
	return isShooting;
}
bool man::GetRush()
{
	return isRush;
}
bool man::GetMovingLeft()
{
	return isMovingLeft;
}

bool man::GetMovingRight()
{
	return isMovingRight;
}

bool man::Getleft()
{
	return left;
}

bool man::Getright()
{
	return right;
}

void man::SetXY(int nx, int ny)
{
	x = nx;
	y = ny;
```

```
}

void man::Setfly(bool flag)
{
	fly = flag;
	if (fly && velocity <= 1) rising = true;
}

void man::jumpsound()
{
	CAudio::Instance()->Play(3,false);
}

void man::DeleteBullet()
{
	////////////////////////////////////////刪除子彈////////////////////////////
	int shootnum = (int)shoot.size();
	for (int i=0; i < shootnum; i++)
	{
		delete shoot[0];
		shoot.erase(shoot.begin());
	}
	/////////////////////////////////////////////////////////////////////////
}

int man::GetBullet()
{
	return shoot.size();
}

int man::GetDamage()
{
	return damage;
}

void man::AddDamage()
{
	damage +=1;
}

void man::SetCombodelay()
{
	combodelay = 5;
}

void man::OnShow()
{
	//////////////////////////////////////////////////////////////////顯示角色//////////////////////////////////////////
	if(animation.GetCurrentBitmapNumber() > 0 && animation.GetCurrentBitmapNumber() < 8) animation.SetTopLeft(x-19, y);//修正座標
	else if (animation.GetCurrentBitmapNumber() > 18 && animation.GetCurrentBitmapNumber() < 23) animation.SetTopLeft(x, y-12);//修正座
	標
	else	animation.SetTopLeft(x, y);
	animation.OnShow();
}
void man::OnShow(int i)
{
	//////////////////////////////////////////////////////////////////顯示角色//////////////////////////////////////////
	animation.SetTopLeft(x, y);
	animation.OnShow();
	//////////////////////////////////////////////////////////////////顯示子彈//////////////////////////////////////////
	if (shoot.size() != 0) {
		for (int count = 0; count < this->GetBullet(); count++) {
			shoot[count]->OnShow();
		}
	}
	//////////////////////////////////////////////////////////////////顯示能量條//////////////////////////////////////////
	MagicBar.SetTopLeft(MagicX, MagicY);
	MagicBar.ShowBitmap();
	for (int i = 0; i < 100 - magic; i++)
	{
		if (characteristic == "ice")
		{
			Black.SetTopLeft(MagicX + (2 * i), MagicY + 1);
			Black.ShowBitmap();
		}
		if (characteristic == "fire")
```

```
                {
                        Black.SetTopLeft(MagicX +200 - (2 * i), MagicY + 1);
                        Black.ShowBitmap();
                }
        }
        ////////////////////////////////////////////////////////////////////////////////////////////////
    }
}
```

bullet.h

```
namespace game_framework {
    class bullet
    {
    public:
        bullet(int _x, int _y,int _direction,string _characteristic);
        int     GetX1();                                    // 角色左上角 x 座標
        int     GetY1();                                    // 角色左上角 y 座標
        int GetX2();
        int GetY2();
        int Getdirection();
        void AddPic();                          // 載入圖形
        void OnShow();
        void Move();
        void SetXY(int _x, int _y);
        CAnimation Getbullet();
    protected:
        CAnimation bulletPic;                   // 利用動畫作圖形
        string characteristic;
        int x, y, direction;
    };

}
#endif
```

bullet.cpp

```
#include "stdafx.h"
#include "Resource.h"
#include <mmsystem.h>
#include <ddraw.h>
#include "audio.h"
#include "gamelib.h"
#include "bullet.h"

namespace game_framework{
    bullet::bullet(int _x, int _y,int _direction,string _characteristic)
    {
        ////////////////////////初始化////////////////////////////////
        if (_direction == 1) _x -= 15;
        else _x += 15;
        SetXY(_x, _y);
        direction = _direction;
        characteristic = _characteristic;
        AddPic();
        ////////////////////////////////////////////////////////////
    }
    int bullet::GetX1()
    {
        return x;
    }
    int bullet::GetY1()
    {
        return y;
    }
    int bullet::GetX2()
    {
        return x+bulletPic.Width();
    }
    int bullet::GetY2()
    {
        return y+bulletPic.Height();
    }
    int bullet::Getdirection()
    {
        return direction;
    }
    void bullet::AddPic()
    {
```

```
	////////////////////////////讀取圖片////////////////////////////
	bulletPic.AddBitmap(BLACK, RGB(0, 0, 0));                                   //0
	bulletPic.AddBitmap(FIREBULLET1, RGB(8, 96, 0));            //0
	bulletPic.AddBitmap(FIREBULLET2, RGB(8, 96, 0));            //1
	bulletPic.AddBitmap(FIREBULLET3, RGB(8, 96, 0));            //2
	bulletPic.AddBitmap(FIREBULLET4, RGB(8, 96, 0));            //3
	bulletPic.AddBitmap(FIREBULLET5, RGB(8, 96, 0));            //4
	bulletPic.AddBitmap(FIREBULLET6, RGB(8, 96, 0));            //5
	bulletPic.AddBitmap(FIREBULLET7, RGB(8, 96, 0));            //6
	bulletPic.AddBitmap(FIREBULLET8, RGB(8, 96, 0));            //7
	bulletPic.AddBitmap(FIREBULLET9, RGB(8, 96, 0));            //8
	bulletPic.AddBitmap(FIREBULLET10, RGB(8, 96, 0));           //9
	bulletPic.AddBitmap(FIREBULLET11, RGB(8, 96, 0));           //10
	bulletPic.AddBitmap(FIREBULLET12, RGB(8, 96, 0));           //11
	bulletPic.AddBitmap(FIREBULLET13, RGB(8, 96, 0));           //12
	bulletPic.AddBitmap(FIREBULLET14, RGB(8, 96, 0));           //13
	bulletPic.AddBitmap(FIREBULLET15, RGB(8, 96, 0));           //14
	bulletPic.AddBitmap(FIREBULLET16, RGB(8, 96, 0));           //15
	bulletPic.AddBitmap(FIREBULLET17, RGB(8, 96, 0));           //16
	bulletPic.AddBitmap(FIREBULLET18, RGB(8, 96, 0));           //17
	bulletPic.AddBitmap(FIREBULLET19, RGB(8, 96, 0));           //18
	bulletPic.AddBitmap(FIREBULLET20, RGB(8, 96, 0));           //19
	bulletPic.AddBitmap(FIREBULLET21, RGB(8, 96, 0));           //20
	bulletPic.AddBitmap(FIREBULLET22, RGB(8, 96, 0));           //21
	bulletPic.AddBitmap(FIREBULLET23, RGB(8, 96, 0));           //22
	bulletPic.AddBitmap(ICEBULLET1, RGB(0, 160, 160));          //23
	bulletPic.AddBitmap(ICEBULLET2, RGB(0, 160, 160));          //24
	bulletPic.AddBitmap(ICEBULLET3, RGB(0, 160, 160));          //25
	bulletPic.AddBitmap(ICEBULLET4, RGB(0, 160, 160));          //26
	bulletPic.AddBitmap(ICEBULLET5, RGB(0, 160, 160));          //27
	bulletPic.AddBitmap(ICEBULLET2, RGB(0, 160, 160));          //28
	bulletPic.AddBitmap(ICEBULLET3, RGB(0, 160, 160));          //29
	bulletPic.AddBitmap(ICEBULLET4, RGB(0, 160, 160));          //30
	bulletPic.AddBitmap(ICEBULLET5, RGB(0, 160, 160));          //31
	bulletPic.AddBitmap(ICEBULLET6, RGB(255, 251, 255));                //32
	bulletPic.AddBitmap(ICEBULLET7, RGB(255, 251, 255));                //33
	bulletPic.AddBitmap(ICEBULLET8, RGB(255, 251, 255));                //34
	bulletPic.AddBitmap(ICEBULLET9, RGB(0, 160, 160));          //35
	bulletPic.AddBitmap(ICEBULLET10, RGB(0, 160, 160));         //36
	bulletPic.AddBitmap(ICEBULLET11, RGB(0, 160, 160));         //37
	bulletPic.AddBitmap(ICEBULLET12, RGB(0, 160, 160));         //38
	bulletPic.AddBitmap(ICEBULLET13, RGB(0, 160, 160));         //39
	bulletPic.AddBitmap(ICEBULLET10, RGB(0, 160, 160));         //40
	bulletPic.AddBitmap(ICEBULLET11, RGB(0, 160, 160));         //41
	bulletPic.AddBitmap(ICEBULLET12, RGB(0, 160, 160));         //42
	bulletPic.AddBitmap(ICEBULLET13, RGB(0, 160, 160));         //43
	bulletPic.AddBitmap(ICEBULLET14, RGB(255, 251, 255));               //44
	bulletPic.AddBitmap(ICEBULLET15, RGB(255, 251, 255));               //45
	bulletPic.AddBitmap(ICEBULLET16, RGB(255, 251, 255));               //46
	bulletPic.SetDelayCount(3);
	bulletPic.Reset();
	//////////////////////////////////////////////////////////////////////////
}
void bullet::SetXY(int _x, int _y) {
	x = _x;
	y = _y;
}
CAnimation bullet::Getbullet()
{
	return bulletPic;
}
void bullet::Move()
{
	////////////////////////////////////子彈移動//////////////////////////////////////////
	if (direction==1)x += 15;
	else x -= 15;
	////////////////////////////////////根據屬性及方向決定圖片////////////////////////////////
	if (direction == 1 && characteristic=="fire")
	{
		if (!(bulletPic.GetCurrentBitmapNumber() >= 1 && bulletPic.GetCurrentBitmapNumber() < 12))
			bulletPic.Movetonum(1);
		else bulletPic.OnMove();
	}
	if (direction == 0 && characteristic == "fire")
	{
		if (!(bulletPic.GetCurrentBitmapNumber() >= 13 && bulletPic.GetCurrentBitmapNumber() < 23))
```

```
                        bulletPic.Movetonum(13);
                    else bulletPic.OnMove();
                }
                if (direction == 1 && characteristic == "ice")
                {
                    if (!(bulletPic.GetCurrentBitmapNumber() >= 24 && bulletPic.GetCurrentBitmapNumber() < 35))
                        bulletPic.Movetonum(24);
                    else bulletPic.OnMove();
                }
                if (direction == 0 && characteristic == "ice")
                {
                    if (!(bulletPic.GetCurrentBitmapNumber() >= 36 && bulletPic.GetCurrentBitmapNumber() < 47))
                        bulletPic.Movetonum(36);
                    else bulletPic.OnMove();
                }
                ////////////////////////////////////////////////////////////////////////////////////////
        }
        void bullet::OnShow()
        {
                bulletPic.SetTopLeft(x, y);
                bulletPic.OnShow();
        }
}
```

iceman.h

```
#ifndef ICEMAN_H
#define ICEMAN_H

#include "man.h"
namespace game_framework {
        class iceman : public man
        {
        public:
                iceman();
                virtual void LoadBitmap();
                virtual void LoadBitmap(int);
                virtual void Initialize();
                virtual void jumpsound();
        };
}
#endif
```

iceman.cpp

```
#include "stdafx.h"
#include "Resource.h"
#include <mmsystem.h>
#include <ddraw.h>
#include "audio.h"
#include "gamelib.h"
#include "iceman.h"

namespace game_framework {
        iceman::iceman()
        {
                Initialize();
        }
        void iceman::LoadBitmap()
        {
                ////////////////////////////////////讀取圖片//////////////////////////////////////
                animation.AddBitmap(ICEMAN, RGB(255, 255, 255));
                animation.AddBitmap(ICERUN01, RGB(255, 255, 255));
                animation.AddBitmap(ICERUN02, RGB(255, 255, 255));
                animation.AddBitmap(ICERUN03, RGB(255, 255, 255));
                animation.AddBitmap(ICERUN04, RGB(255, 255, 255));
                animation.AddBitmap(ICERUN05, RGB(255, 255, 255));
                animation.AddBitmap(ICERUN06, RGB(255, 255, 255));
                animation.AddBitmap(ICERUN07, RGB(255, 255, 255));
                animation.AddBitmap(ICERUN08, RGB(255, 255, 255));
                animation.AddBitmap(ICERUN09, RGB(255, 255, 255));
                animation.AddBitmap(ICERUN10, RGB(255, 255, 255));
                animation.AddBitmap(ICERUN11, RGB(255, 255, 255));
                animation.AddBitmap(ICERUN12, RGB(255, 255, 255));
                animation.AddBitmap(ICERUN13, RGB(255, 255, 255));
                animation.AddBitmap(ICERUN14, RGB(255, 255, 255));
                animation.AddBitmap(ICEJUMP00, RGB(255, 255, 255));
                animation.AddBitmap(ICEJUMP01, RGB(255, 255, 255));
                animation.AddBitmap(ICEJUMP02, RGB(255, 255, 255));
```

```
	animation.AddBitmap(ICEJUMP03, RGB(255, 255, 255));
	animation.AddBitmap(ICEDOWN00, RGB(255, 255, 255));
	animation.AddBitmap(ICEDOWN01, RGB(255, 255, 255));
	animation.AddBitmap(ICEDOWN02, RGB(255, 255, 255));
	animation.AddBitmap(ICEDOWN03, RGB(255, 255, 255));
	//////////////////////////////////////////////////////////////
}
void iceman::LoadBitmap(int i)
{
	/////////////////////////////////////讀取對戰冰人圖片//////////////////////
	animation.AddBitmap(IceAttack, RGB(0, 0, 0));				//0
	animation.AddBitmap(ICEATTACK2, RGB(0, 0, 0));					//1
	animation.AddBitmap(ICEATTACKRUN1, RGB(0, 0, 0));			//2
	animation.AddBitmap(ICEATTACKRUN1, RGB(0, 0, 0));			//3
	animation.AddBitmap(ICEATTACKRUN2, RGB(0, 0, 0));			//4
	animation.AddBitmap(ICEATTACKRUN2, RGB(0, 0, 0));			//5
	animation.AddBitmap(ICEATTACKRUN3, RGB(0, 0, 0));			//6
	animation.AddBitmap(ICEATTACKRUN3, RGB(0, 0, 0));			//7
	animation.AddBitmap(ICEATTACKRUN4, RGB(0, 0, 0));			//8
	animation.AddBitmap(ICEATTACKRUN4, RGB(0, 0, 0));			//9
	animation.AddBitmap(ICEATTACKRUN5, RGB(0, 0, 0));			//10
	animation.AddBitmap(ICEATTACKRUN5, RGB(0, 0, 0));			//11
	animation.AddBitmap(ICEATTACKRUN6, RGB(0, 0, 0));			//12
	animation.AddBitmap(ICEATTACKRUN6, RGB(0, 0, 0));			//13
	animation.AddBitmap(ICEATTACKJUMP1, RGB(0, 0, 0));				//14
	animation.AddBitmap(ICEATTACKJUMP2, RGB(0, 0, 0));				//15
	animation.AddBitmap(ICEATTACKED1, RGB(0, 0, 0));			//16
	animation.AddBitmap(ICEATTACKED2, RGB(0, 0, 0));			//17
	animation.AddBitmap(ICEATTACKED3, RGB(0, 0, 0));			//18
	animation.AddBitmap(ICEATTACKED4, RGB(0, 0, 0));			//19
	animation.AddBitmap(ICEATTACKED5, RGB(0, 0, 0));			//20
	animation.AddBitmap(ICEATTACKED6, RGB(0, 0, 0));			//21
	animation.AddBitmap(ICEATTACKED7, RGB(0, 0, 0));			//22
	animation.AddBitmap(ICEATTACKED8, RGB(0, 0, 0));			//23
	animation.AddBitmap(ICEATTACKED9, RGB(0, 0, 0));			//24
	animation.AddBitmap(ICEATTACKED10, RGB(0, 0, 0));			//25
	animation.AddBitmap(ICEATTACKED11, RGB(0, 0, 0));			//26
	animation.AddBitmap(ICEATTACKED12, RGB(0, 0, 0));			//27
	animation.AddBitmap(ICE_HIT1, RGB(0, 0, 0));				//28
	animation.AddBitmap(ICE_HIT1, RGB(0, 0, 0));				//29
	animation.AddBitmap(ICE_HIT2, RGB(0, 0, 0));				//30
	animation.AddBitmap(ICE_HIT2, RGB(0, 0, 0));				//31
	animation.AddBitmap(ICE_HIT3, RGB(0, 0, 0));				//32
	animation.AddBitmap(ICE_HIT3, RGB(0, 0, 0));				//33
	animation.AddBitmap(ICE_HIT4, RGB(0, 0, 0));				//34
	animation.AddBitmap(ICE_HIT4, RGB(0, 0, 0));				//35
	animation.AddBitmap(ICE_DEFENSE1, RGB(0, 0, 0));			//36
	animation.AddBitmap(ICE_DEFENSE2, RGB(0, 0, 0));			//37
	animation.AddBitmap(ICEATTACKMAGIC, RGB(0, 0, 0));				//38
	animation.AddBitmap(ICEATTACKMAGIC3, RGB(0, 0, 0));				//39
	animation.AddBitmap(ICEATTACKMAGIC2, RGB(0, 0, 0));				//40
	animation.AddBitmap(ICEATTACKMAGIC4, RGB(0, 0, 0));				//41
	/////////////////////////////////////魔力條及魔力擋板//////////////////////////
	MagicBar.LoadBitmap(MAGICBAR);
	Black.LoadBitmap(BLACK);
	//////////////////////////////////////////////////////////////////////////////
	animation.OnMove();
}
void iceman::Initialize()
{
	/////////////////////////////////////初始化////////////////////////////////
	floor = 460;
	const int X_POS = 60;
	const int Y_POS = 400 - 39;
	const int INITIAL_VELOCITY = 12;
	const int STEP_SIZE = 7;
	step_size = 7;
	initial_velocity = INITIAL_VELOCITY;
	velocity = initial_velocity;
	characteristic = "ice";
	gem = 0;
	x = X_POS;
	y = Y_POS;
	times = 0;
	direction = 0;
	isAttacked = 2;
```

```
		combodelay = 0;
		damage = 0;
		RushTime = 0;
		magic = 100;
		MagicX = 364;
		MagicY = 30;
		isMagic=isRush=isShooting = isAttack = isTwojump = fly = rising = isMovingLeft = isMovingRight = isMovingUp = isMovingDown = false;
		////////////////////////////////////////////////////////////////////////////////
	}
	void iceman::jumpsound()
	{
		CAudio::Instance()->Play(3,false);//跳躍聲音
	}
}
```

fireman.h

```
#ifndef FIREMAN_H
#define FIREMAN_H

#include "man.h"
namespace game_framework {
	class fireman : public man
	{
	public:
		fireman();
		virtual void LoadBitmap();                    // 載入圖形
		virtual void LoadBitmap(int i);
		virtual void Initialize();
		virtual void jumpsound();
	};
}
#endif // !FIREMAN_H
```

fireman.cpp

```
#include "stdafx.h"
#include "Resource.h"
#include <mmsystem.h>
#include <ddraw.h>
#include "audio.h"
#include "gamelib.h"
#include "fireman.h"
namespace game_framework {
	fireman::fireman()
	{
		Initialize();
	}
	void fireman::LoadBitmap()
	{
		//////////////////////////////////////讀取圖片//////////////////////////////////////
		animation.AddBitmap(FIRE, RGB(255, 255, 255));
		animation.AddBitmap(FIREMANRUN01, RGB(255, 255, 255));
		animation.AddBitmap(FIREMANRUN02, RGB(255, 255, 255));
		animation.AddBitmap(FIREMANRUN03, RGB(255, 255, 255));
		animation.AddBitmap(FIREMANRUN04, RGB(255, 255, 255));
		animation.AddBitmap(FIREMANRUN05, RGB(255, 255, 255));
		animation.AddBitmap(FIREMANRUN06, RGB(255, 255, 255));
		animation.AddBitmap(FIREMANRUN07, RGB(255, 255, 255));
		animation.AddBitmap(FIREMANRUN08, RGB(255, 255, 255));
		animation.AddBitmap(FIREMANRUN09, RGB(255, 255, 255));
		animation.AddBitmap(FIREMANRUN10, RGB(255, 255, 255));
		animation.AddBitmap(FIREMANRUN11, RGB(255, 255, 255));
		animation.AddBitmap(FIREMANRUN12, RGB(255, 255, 255));
		animation.AddBitmap(FIREMANRUN13, RGB(255, 255, 255));
		animation.AddBitmap(FIREMANRUN14, RGB(255, 255, 255));
		animation.AddBitmap(FIREJUMP00, RGB(255, 255, 255));
		animation.AddBitmap(FIREJUMP01, RGB(255, 255, 255));
		animation.AddBitmap(FIREJUMP02, RGB(255, 255, 255));
		animation.AddBitmap(FIREJUMP03, RGB(255, 255, 255));
		animation.AddBitmap(FIREDOWN00, RGB(255, 255, 255));
		animation.AddBitmap(FIREDOWN01, RGB(255, 255, 255));
		animation.AddBitmap(FIREDOWN02, RGB(255, 255, 255));
		animation.AddBitmap(FIREDOWN03, RGB(255, 255, 255));
		////////////////////////////////////////////////////////////////////////////////
	}
	void fireman::LoadBitmap(int i)
	{
		//////////////////////////////////////讀取對戰模式圖片//////////////////////////////
		animation.AddBitmap(FIRE_ATTTACK, RGB(0, 0, 0));                    //0
```

```
		animation.AddBitmap(FIRE_ATTTACK2, RGB(0, 0, 0));			//1
		animation.AddBitmap(FIREATTACKRUN1, RGB(0, 0, 0));				//2
		animation.AddBitmap(FIREATTACKRUN1, RGB(0, 0, 0));				//3
		animation.AddBitmap(FIREATTACKRUN2, RGB(0, 0, 0));				//4
		animation.AddBitmap(FIREATTACKRUN2, RGB(0, 0, 0));				//5
		animation.AddBitmap(FIREATTACKRUN3, RGB(0, 0, 0));				//6
		animation.AddBitmap(FIREATTACKRUN3, RGB(0, 0, 0));				//7
		animation.AddBitmap(FIREATTACKRUN4, RGB(0, 0, 0));				//8
		animation.AddBitmap(FIREATTACKRUN4, RGB(0, 0, 0));				//9
		animation.AddBitmap(FIREATTACKRUN5, RGB(0, 0, 0));				//10
		animation.AddBitmap(FIREATTACKRUN5, RGB(0, 0, 0));				//11
		animation.AddBitmap(FIREATTACKRUN6, RGB(0, 0, 0));				//12
		animation.AddBitmap(FIREATTACKRUN6, RGB(0, 0, 0));				//13
		animation.AddBitmap(FIREATTTACKEDJUMP1, RGB(0, 0, 0));		//14
		animation.AddBitmap(FIREATTTACKEDJUMP2, RGB(0, 0, 0));		//15
		animation.AddBitmap(FIRE_ATTTACKED1, RGB(0, 0, 0));				//16
		animation.AddBitmap(FIRE_ATTTACKED2, RGB(0, 0, 0));				//17
		animation.AddBitmap(FIRE_ATTTACKED3, RGB(0, 0, 0));				//18
		animation.AddBitmap(FIRE_ATTTACKED4, RGB(0, 0, 0));				//19
		animation.AddBitmap(FIRE_ATTTACKED5, RGB(0, 0, 0));				//20
		animation.AddBitmap(FIRE_ATTTACKED6, RGB(0, 0, 0));				//21
		animation.AddBitmap(FIRE_ATTTACKED7, RGB(0, 0, 0));				//22
		animation.AddBitmap(FIRE_ATTTACKED8, RGB(0, 0, 0));				//23
		animation.AddBitmap(FIRE_ATTTACKED9, RGB(0, 0, 0));				//24
		animation.AddBitmap(FIRE_ATTTACKED10, RGB(0, 0, 0));			//25
		animation.AddBitmap(FIRE_ATTTACKED11, RGB(0, 0, 0));			//26
		animation.AddBitmap(FIRE_ATTTACKED12, RGB(0, 0, 0));			//27
		animation.AddBitmap(FIRE_HIT1, RGB(0, 0, 0));			//28
		animation.AddBitmap(FIRE_HIT1, RGB(0, 0, 0));			//29
		animation.AddBitmap(FIRE_HIT2, RGB(0, 0, 0));			//30
		animation.AddBitmap(FIRE_HIT2, RGB(0, 0, 0));			//31
		animation.AddBitmap(FIRE_HIT3, RGB(0, 0, 0));			//32
		animation.AddBitmap(FIRE_HIT3, RGB(0, 0, 0));			//33
		animation.AddBitmap(FIRE_HIT4, RGB(0, 0, 0));			//34
		animation.AddBitmap(FIRE_HIT4, RGB(0, 0, 0));			//35
		animation.AddBitmap(FIRE_DEFENSE1, RGB(0, 0, 0));		//36
		animation.AddBitmap(FIRE_DEFENSE2, RGB(0, 0, 0));		//37
		animation.AddBitmap(FIREATTACKMAGIC, RGB(0, 0, 0));				//38
		animation.AddBitmap(FIREATTACKMAGIC3, RGB(0, 0, 0));			//39
		animation.AddBitmap(FIREATTACKMAGIC2, RGB(0, 0, 0));			//40
		animation.AddBitmap(FIREATTACKMAGIC4, RGB(0, 0, 0));			//41
		//////////////////////////////////////魔力條及魔力擋板//////////////////////////
		MagicBar.LoadBitmap(MAGICBAR);
		Black.LoadBitmap(BLACK);
		//////////////////////////////////////////////////////////////////////////////
}
void fireman::Initialize()
{
		//////////////////////////////////////初始化//////////////////////////////
		floor = 460;
		const int X_POS = 60;
		const int Y_POS = 460 - 39;
		const int INITIAL_VELOCITY = 12;
		const int STEP_SIZE = 7;
		step_size = 7;
		gem = 0;
		initial_velocity = INITIAL_VELOCITY;
		velocity = initial_velocity;
		characteristic = "fire";
		x = X_POS;
		y = Y_POS;
		times = 0;
		direction = 1;
		isAttacked = 2;
		combodelay = 0;
		damage = 0;
		RushTime = 0;
		magic = 100;
		MagicX = 76;
		MagicY = 30;
		isMagic =isRush=isShooting = isAttack = isTwojump = fly = rising = isMovingLeft = isMovingRight = isMovingUp = isMovingDown = false;
		//////////////////////////////////////////////////////////////////////////////
}
void fireman::jumpsound()
{
		CAudio::Instance()->Play(4,false);//跳躍聲音
```

```
	}
}
```

elevator.h

```
namespace game_framework
{
	class buttonelevator
	{
	public:
		buttonelevator();
		int GetX();
		int GetY();
		int getox();
		int getoy();
		void setoxy(int x, int y);
		void LoadBitmap();
		void SetTopLeft(int _x, int _y);
		void ShowBitmap();
		int Getvelocity();
		int down();
		int up();
	private:
		int x, y;
		int ox, oy, velocity;
		CMovingBitmap image;
	};
}
```

elevator.cpp

```
#include "stdafx.h"
#include "Resource.h"
#include <mmsystem.h>
#include <ddraw.h>
#include "audio.h"
#include "gamelib.h"
#include "elevator.h"
namespace game_framework
{
	buttonelevator::buttonelevator()
	{
		velocity = 5;
	}
	int buttonelevator::GetX()
	{
		return x;
	}
	int buttonelevator::GetY()
	{
		return y;
	}
	int buttonelevator::getox()
	{
		return ox;
	}
	int buttonelevator::getoy()
	{
		return oy;
	}
	void buttonelevator::setoxy(int x, int y)
	{
		ox = x;
		oy = y;
	}
	void buttonelevator::LoadBitmap()
	{
		image.LoadBitmap(elevator,RGB(255,255,255));
	}
	void buttonelevator::SetTopLeft(int _x, int _y)
	{
		x = _x;
		y = _y;
		image.SetTopLeft(x, y);
	}
	void buttonelevator::ShowBitmap()
	{
		image.ShowBitmap();
	}
```

```
	int buttonelevator::Getvelocity()
	{
		return velocity;
	}
	int buttonelevator::down()
	{
		y += velocity;
		return y;
	}
	int buttonelevator::up()
	{
		y -= velocity;
		return y;
	}
}
```

yellowbutton.h

```
#ifndef YELLOWBUTTON_H
#define YELLOWBUTTON_H
namespace game_framework
{
	class yellowbutton
	{
	public:
		yellowbutton();
		int GetX();
		int	GetY();
		bool Onpress(int _x,int _y);
		virtual void LoadBitmap();
		void SetTopLeft(int _x, int _y);
		void ShowBitmap();
	protected:
		int x, y;
		CMovingBitmap image;
	};

}
#endif
```

yellowbutton.cpp

```
#include "stdafx.h"
#include "Resource.h"
#include <mmsystem.h>
#include <ddraw.h>
#include "audio.h"
#include "gamelib.h"
#include "yellowbutton.h"
namespace game_framework
{
	yellowbutton::yellowbutton()
	{

	}
	int yellowbutton::GetX()
	{
		return x;
	}
	int yellowbutton::GetY()
	{
		return y;
	}
	bool yellowbutton::Onpress(int _x, int _y)
	{
		//////////////////////////// 角色站在按鈕上//////////////////////
		if (x < _x && _x < x + 20 && y < _y && _y < y + 20) return true;
		else return false;
	}
	void yellowbutton::LoadBitmap()
	{
		image.LoadBitmap(Button,RGB(255, 255, 255));
	}
	void yellowbutton::SetTopLeft(int _x, int _y)
	{
		x = _x;
		y = _y;
		image.SetTopLeft(x, y+13);
	}
	void yellowbutton::ShowBitmap()
```

```
	{
		image.ShowBitmap();
	}
}
```

redbutton.h

```
#include "yellowbutton.h"
namespace game_framework
{
	class RedButton : public yellowbutton
	{
	public:
		RedButton();
		virtual void LoadBitmap();
	};
}
```

redbutton.cpp

```
#include "stdafx.h"
#include "Resource.h"
#include <mmsystem.h>
#include <ddraw.h>
#include "audio.h"
#include "gamelib.h"
#include "RedButton.h"

namespace game_framework {
	RedButton::RedButton()
	{
	}
	void RedButton::LoadBitmap()
	{
		image.LoadBitmap(Button, RGB(255, 255, 255));
	}
}
```

Stone.h

```
#include "man.h"
namespace game_framework
{
	class rockStone {
		public:
			rockStone();
			int	GetX();				// 石頭左上角 x 座標
			int	GetY();				// 石頭左上角 y 座標
			int	GetX2();			// 石頭右下角 x 座標
			int	GetY2();			// 石頭右下角 y 座標
			void LoadBitmap();
			void Down();
			bool CheckStone(int ,int,int,int);
			void OnMove();
			void SetTopLeft(int _x, int _y);
			void ShowBitmap();
			void SetXY(int _x, int _y);
	private:
			int x, y, x2 ,y2;
			CMovingBitmap image;
	};
}
```

Stone.cpp

```
#include "stdafx.h"
#include "Resource.h"
#include <mmsystem.h>
#include <ddraw.h>
#include "audio.h"
#include "gamelib.h"
#include"stone.h"
namespace game_framework
{
	rockStone::rockStone()
	{
		x = y = x2 = y2 = 0;
	}
	int rockStone::GetX()
	{
		return x;
	}
	int rockStone::GetX2()
	{
```

```
		x2 = x + 39;
		return x2;
	}
	int rockStone::GetY()
	{
		return y;
	}
	int rockStone::GetY2()
	{
		y2 = y + 39;
		return y2;
	}
	void rockStone::SetXY(int _x, int _y)
	{
		x = _x;
		y = _y;
	}
	void rockStone::LoadBitmap()
	{
		image.LoadBitmap(Stone, RGB(255, 255, 255));
	}
	void rockStone::Down()
	{
		y += 1;//石頭下墜
	}
	bool rockStone::CheckStone(int px1,int px2,int py1,int py2)
	{
		////////////////////////////////////若是角色碰種到石頭//////////////////////////
		if ((px1 > GetX() && px1 < GetX2() &&py1 > GetY() && py1 < GetY2())) return false;
		if ((px2 > GetX() && px2 < GetX2() &&py2 > GetY() && py2 < GetY2()))return false;
		if ((px1 > GetX() && px1 < GetX2() &&py2 > GetY() && py2 < GetY2()))return false;
		if ((px2 > GetX() && px2 < GetX2() &&py1 > GetY() && py1 < GetY2()))return false;
		return true;
	}
	void rockStone::ShowBitmap()
	{
		image.SetTopLeft(x, y);
		image.ShowBitmap();
	}
}
```

door.h

```
#ifndef DOOR_H
#define DOOR_H
#include "man.h"
namespace game_framework {
	class door
	{
	public:
		door();
		int		GetX1();				// 角色左上角 x 座標
		int		GetY1();				// 角色左上角 y 座標
		CAnimation* GetDoor();
		virtual void LoadBitmap();			// 載入圖形
		void OnShow(int _x,int _y);
		virtual bool Clearance(man player,int gemnum);
	protected:
		CAnimation animation;			// 利用動畫作圖形
		int x, y;
	};
}
#endif
```

door.cpp

```
#include "stdafx.h"
#include "Resource.h"
#include <mmsystem.h>
#include <ddraw.h>
#include "audio.h"
#include "gamelib.h"
#include "door.h"
namespace game_framework {
	door::door()
	{
	}
	int door::GetX1()
	{
		return x;
```

```
	}
	int door::GetY1()
	{
		return y;
	}
	CAnimation* door::GetDoor()
	{
		return &animation;
	}
	void door::LoadBitmap()
	{
		return ;
	}
	void door::OnShow(int _x,int _y)
	{
		x = _x; y = _y;
		animation.SetTopLeft(x, y);
		animation.OnShow();
	}
	bool door::Clearance(man player,int gemnum)
	{
		return false;
	}
}
```

icedoor.h

```
#ifndef ICEDOOR_H
#define ICEDOOR_H

#include "door.h"
namespace game_framework {
	class icedoor : public door
	{
	public:
		icedoor();
		virtual void LoadBitmap();
		virtual bool Clearance(man player, int icegemnum);
	};

}
#endif
```

icedoor.cpp

```
#include "stdafx.h"
#include "Resource.h"
#include <mmsystem.h>
#include <ddraw.h>
#include "audio.h"
#include "gamelib.h"
#include "icedoor.h"
namespace game_framework {
	icedoor::icedoor()
	{
	}
	void icedoor::LoadBitmap()
	{
		animation.AddBitmap(ICEDOOR1, RGB(0, 0, 0));
		animation.AddBitmap(ICEDOOR2, RGB(0, 0, 0));
		animation.AddBitmap(ICEDOOR3, RGB(0, 0, 0));
		animation.AddBitmap(ICEDOOR4, RGB(0, 0, 0));
		animation.AddBitmap(ICEDOOR5, RGB(0, 0, 0));
		animation.AddBitmap(ICEDOOR6, RGB(0, 0, 0));
		animation.SetDelayCount(3);
	}
	bool icedoor::Clearance(man player, int icegemnum)
	{
		////////////////////////////////////若是為冰人//////////////////////////////////
		if (player.Getcharacteristic() == "ice" && player.getgem() == icegemnum)
		{
			if (animation.IsFinalBitmap())//判斷門是否打開完成
			{
				return true;//冰人已通關
			}
			else
			{
				animation.OnMove();//將門打開
				return false;
```

```
                }
            }
            else
            {
                if (animation.GetCurrentBitmapNumber() != 0)
                {
                    animation.Movetonum(animation.GetCurrentBitmapNumber() - 1);//將門關上
                }
                return false;
            }
        }
};
```

firedoor.h

```
#ifndef FIREDOOR_H
#define FIREDOOR_H

#include "door.h"
namespace game_framework {
    class firedoor : public door
    {
    public:
        firedoor();
        virtual void LoadBitmap();
        virtual bool Clearance(man player, int firegemnum);
    };

}
#endif
```

firedoor.cpp

```
#include "stdafx.h"
#include "Resource.h"
#include <mmsystem.h>
#include <ddraw.h>
#include "audio.h"
#include "gamelib.h"
#include "firedoor.h"
namespace game_framework {
    firedoor::firedoor()
    {
    }
    void firedoor::LoadBitmap()
    {
        animation.AddBitmap(FIREDOOR1, RGB(0, 0, 0));
        animation.AddBitmap(FIREDOOR2, RGB(0, 0, 0));
        animation.AddBitmap(FIREDOOR3, RGB(0, 0, 0));
        animation.AddBitmap(FIREDOOR4, RGB(0, 0, 0));
        animation.AddBitmap(FIREDOOR5, RGB(0, 0, 0));
        animation.AddBitmap(FIREDOOR6, RGB(0, 0, 0));
        animation.AddBitmap(FIREDOOR6, RGB(0, 0, 0));
        animation.SetDelayCount(3);
    }
    bool firedoor::Clearance(man player, int firegemnum)
    {
        ////////////////////////////////////若是為火人//////////////////////////////////////
        if (player.Getcharacteristic() == "fire" && player.getgem() == firegemnum)
        {
            if (animation.IsFinalBitmap())//判斷門是否打開完成
            {
                return true;//火人已通關
            }
            else
            {
                animation.OnMove();//將門打開
                return false;
            }
        }
        else
        {
            if (animation.GetCurrentBitmapNumber() != 0)
            {
                animation.Movetonum(animation.GetCurrentBitmapNumber() - 1);//將門關上
            }
            return false;
        }
    }
```

```
};
```

trap.h

```
#ifndef TRAP_H
#define TRAP_H
//繼承用//
namespace game_framework {
	class trap
	{
	public:
		trap();
		int		GetX1();				// 角色左上角 x 座標
		int		GetY1();				// 角色左上角 y 座標
		virtual void LoadBitmap();			// 載入圖形
		virtual void SetDelayTime()=0;
		void OnMove();
		void OnShow(int _x, int _y);
	protected:
		CAnimation animation;
		int x, y;
	};
}
#endif
```

trap.cpp

```
#include "stdafx.h"
#include "Resource.h"
#include <mmsystem.h>
#include <ddraw.h>
#include "audio.h"
#include "gamelib.h"
#include "door.h"
#include "trap.h"
namespace game_framework {
	trap::trap()
	{
	}
	int trap::GetX1()
	{
		return x;
	}
	int trap::GetY1()
	{
		return y;
	}
	void trap::LoadBitmap()
	{
		return;
	}
	void trap::OnMove()
	{
		animation.OnMove();
	}
	void trap::OnShow(int _x, int _y)
	{
		x = _x; y = _y;
		animation.SetTopLeft(x, y);
		animation.OnShow();
	}
};
```

fire.h

```
#ifndef FIRE_H
#define FIRE_H

#include "trap.h"
namespace game_framework {
	class fire : public trap
	{
	public:
		fire();
		virtual void LoadBitmap();
		virtual void SetDelayTime();
	};
}
#endif
```

fire.cpp

```
#include "stdafx.h"
```

```
#include "Resource.h"
#include <mmsystem.h>
#include <ddraw.h>
#include "audio.h"
#include "gamelib.h"
#include "fire.h"
namespace game_framework {
	fire::fire()
	{
	}
	void fire::LoadBitmap()
	{
		animation.AddBitmap(FIRE1, RGB(0, 0, 0));
		animation.AddBitmap(FIRE2, RGB(0, 0, 0));
		animation.AddBitmap(FIRE3, RGB(0, 0, 0));
		animation.AddBitmap(FIRE4, RGB(0, 0, 0));
		animation.AddBitmap(FIRE5, RGB(0, 0, 0));
		animation.SetDelayCount(5);
	}
	void fire::SetDelayTime()
	{
		animation.SetDelayCount(5);
		animation.Reset();
	}
}
```

water.h

```
#ifndef WATER_H
#define WATER_H
#include "trap.h"
namespace game_framework {
	class water : public trap
	{
	public:
		water();
		virtual void LoadBitmap();
		virtual void SetDelayTime();
	};
}
#endif
```

water.cpp

```
#include "stdafx.h"
#include "Resource.h"
#include <mmsystem.h>
#include <ddraw.h>
#include "audio.h"
#include "gamelib.h"
#include "water.h"
namespace game_framework {
	water::water()
	{
	}
	void water::LoadBitmap()
	{
		animation.AddBitmap(WATER1, RGB(0, 0, 0));
		animation.AddBitmap(WATER2, RGB(0, 0, 0));
		animation.AddBitmap(WATER3, RGB(0, 0, 0));
		animation.AddBitmap(WATER4, RGB(0, 0, 0));
		animation.AddBitmap(WATER5, RGB(0, 0, 0));
		animation.SetDelayCount(3);
	}
	void water::SetDelayTime()
	{
		animation.SetDelayCount(3);
		animation.Reset();
	}
}
```

poison.h

```
#pragma once
#ifndef POISON_H
#define POISON_H
#include "trap.h"
namespace game_framework {
	class poison : public trap
	{
	public:
		poison();
```

```
		virtual void LoadBitmap();
		virtual void SetDelayTime();
	};

}
#endif
```

poison.cpp

```
#include "stdafx.h"
#include "Resource.h"
#include <mmsystem.h>
#include <ddraw.h>
#include "audio.h"
#include "gamelib.h"
#include "poison.h"
namespace game_framework {
	poison::poison()
	{
	}
	void poison::LoadBitmap()
	{
		animation.AddBitmap(POISON1, RGB(30, 30, 30));
		animation.AddBitmap(POISON2, RGB(30, 30, 30));
		animation.AddBitmap(POISON3, RGB(30, 30, 30));
		animation.AddBitmap(POISON4, RGB(30, 30, 30));
		animation.AddBitmap(POISON5, RGB(30, 30, 30));
		animation.SetDelayCount(5);
	}
	void poison::SetDelayTime()
	{
		animation.SetDelayCount(5);
		animation.Reset();
	}
}
```

Rising.h

```
#include "man.h"
namespace game_framework
{
	class Rising {
	public:
		Rising();
		int getX();
		int getY();
		void SetXY(int _x,int _y);
		bool SetRising(man *player);
		void LoadBitmap();
		void OnShow();
	private:
		int x, y;
		CMovingBitmap image;
		CAnimation smoke;
	};
}
```

Rising.cpp

```
#include "stdafx.h"
#include "Resource.h"
#include <mmsystem.h>
#include <ddraw.h>
#include "audio.h"
#include "gamelib.h"
#include "Rising.h"
namespace game_framework
{
	Rising::Rising()
	{
		x = 0,y = 0;
	}
	int Rising::getX()
	{
		return x;
	}
	int Rising::getY()
	{
		return y;
	}
	void Rising::SetXY(int _x,int _y)
	{
```

```
		x = _x;
		y = _y-5;
	}
	bool Rising::SetRising(man * player)
	{
		//////////////////////////////////////////////////判斷站在噴射器上//////////////////////////////////////////////////////////
		if (((player->GetX1() < x + image.Width() && player->GetX1() > x) || (player->GetX2() < x + image.Width() && player->GetX2() > x)) &&
		(player->GetY2() < y&&player->GetY1()>y-160)) return true;
		else return false;
	}
	void Rising::LoadBitmap()
	{
		image.LoadBitmap(pic_rising, RGB(0, 0, 0));
		smoke.AddBitmap(SMOKE, RGB(0, 0, 0));
		smoke.AddBitmap(SMOKE2, RGB(0, 0, 0));
		smoke.AddBitmap(SMOKE3, RGB(0, 0, 0));
	}
	void Rising::OnShow()
	{
		image.SetTopLeft(x, y);
		smoke.SetTopLeft(x, y - 160);
		image.ShowBitmap();
		smoke.OnShow();
		smoke.OnMove();
	}
}
```

FallWall.h

```
#pragma once
#ifndef FALLWALL_H
#define FALLWALL_H
#include "man.h"
namespace game_framework
{
	class FallWall
	{
	public:
		FallWall();
		virtual void LoadBitmap();
		int GetX();
		int GetY();
		int Getox();
		int Getoy();
		int GetFall();
		bool Onstand(man player);
		void SetFall(int flag);
		void SetFall(man player);
		void Setoxy(int x, int y);
		void SetTopLeft(int _x, int _y);
		void ShowBitmap();
		void Down();
		void Up();
		void Up(int v);
	protected:
		int x, y;
		int ox, oy;
		int Fall;
		CMovingBitmap image;
	};
}
#endif
```

FallWall.cpp

```
#include "stdafx.h"
#include "Resource.h"
#include <mmsystem.h>
#include <ddraw.h>
#include "audio.h"
#include "gamelib.h"
#include "FallWall.h"

namespace game_framework
{
	FallWall::FallWall()
	{
	}
	void FallWall::LoadBitmap()
	{
```

```
		image.LoadBitmap(elevator, RGB(255, 255, 255));
	}
	int FallWall::GetX()
	{
		return x;
	}
	int FallWall::GetY()
	{
		return y;
	}
	int FallWall::Getox()
	{
		return ox;
	}
	int FallWall::Getoy()
	{
		return oy;
	}
	int FallWall::GetFall()
	{
		return Fall;
	}
	bool FallWall::Onstand(man player)
	{
		////////////////////////////////////////////////////判斷角色站在下落地板上////////////////////////////////////
		if (x < (player.GetX1() + player.GetX2()) / 2 && (player.GetX1() + player.GetX2()) / 2 < x + 60 && (y + 44 > player.GetY2() && y - 5 <
		player.GetY2())) return true;
		else return false;
		//////////////////////////////////////////////////////////////////////////////////////////////////////////////
	}
	void FallWall::SetFall(int flag)
	{
		Fall = flag;
	}
	void FallWall::SetFall(man player)
	{
		if (x < (player.GetX1() + player.GetX2()) / 2 && (player.GetX1() + player.GetX2()) / 2 < x + 60 && (y+40 > player.GetY2() && y-3 <
		player.GetY2()))	Fall+=1;
	}
	void FallWall::Setoxy(int x, int y)
	{
		ox = x;
		oy = y;
	}
	void FallWall::SetTopLeft(int _x, int _y)
	{
		x = _x; y = _y;
	}
	void FallWall::ShowBitmap()
	{
		image.SetTopLeft(x, y);
		image.ShowBitmap();
	}
	void FallWall::Down()
	{
		y+=4;
	}
	void FallWall::Up()
	{
		y--;
	}
	void FallWall::Up(int v)
	{
		y-=v;
	}
}
```

ChainElevator.h

```
#ifndef CHAINELEVATOR_H
#define CHAINELEVATOR_H
#include "FallWall.h"
namespace game_framework
{
	class ChainElevator :public FallWall
	{
	public:
		ChainElevator();
```

```
		void SetX2Y2(int X2,int Y2);
		void Setaimy1(int y);
		void Setaimy2(int y);
		bool Onstand2(man player);
		bool OnStand(int x,int y);
		bool OnStand2(int x,int y);
		int GetFall2();
		void SetFall2(int flag);
		void SetFall2(man player);
		double Getcount();
		int GetaimY1();
		int GetX2();
		int GetY2();
		void LoadBitmap();
		void onShow();
	private:
		int x2,y2;
		int rangey1, rangey2;
		int aimy1, aimy2;
		int Fall2;
		double count;
		CMovingBitmap Elevator2;
	};
}
#endif
```

ChainElevator.cpp

```
#include "stdafx.h"
#include "Resource.h"
#include <mmsystem.h>
#include <ddraw.h>
#include "audio.h"
#include "gamelib.h"
#include "Chainelevator.h"
#include "math.h"
namespace game_framework
{
	ChainElevator::ChainElevator()
	{
	}
	void ChainElevator::SetX2Y2(int X2, int Y2)
	{
		x2 = X2, y2 = Y2;
	}
	void ChainElevator::Setaimy1(int _y)
	{
		aimy1 = _y+10;
		rangey1 = aimy1-y;
	}
	void ChainElevator::Setaimy2(int y)
	{
		aimy2 = y;
		rangey2 = y-y2;
	}
	bool ChainElevator::Onstand2(man player)
	{
		//////////////////////////若是站在電梯 2 上//////////////////////////////////
		if (x2 < (player.GetX1() + player.GetX2()) / 2 && (player.GetX1() + player.GetX2()) / 2 < x2 + 60 && (y2 + 44 +(int)Getcount() >
		player.GetY2() && y2 - 5 + (int)Getcount() < player.GetY2()))
			return true;
		else return false;
	}
	bool ChainElevator::OnStand(int _x, int _y)
	{
		//////////////////////////站在電梯 1 上//////////////////////////////////
		if (x < _x && _x < x + 60 && y + 50		> _y && y - 20		< _y)
			return true;
		else return false;
	}
	bool ChainElevator::OnStand2(int x, int y)
	{
		//////////////////////////站在電梯 2 上//////////////////////////
		if (x2 < x && x < x2 + 60 && (y2 + 44 + (int)Getcount() > y && y2 - 5 + (int)Getcount() < y))
			return true;
		else return false;
	}
	int ChainElevator::GetFall2()
```

```
	{
		return Fall2;
	}
	void ChainElevator::SetFall2(int flag)
	{
		Fall2 = flag;
	}
	void ChainElevator::SetFall2(man player)
	{
		if (x2 < (player.GetX1() + player.GetX2()) / 2 && (player.GetX1() + player.GetX2()) / 2 < x2 + 60 && (y2 + 44 + (int)Getcount() >
		player.GetY2() && y2 - 5 + (int)Getcount() < player.GetY2()))
			Fall2+=1;
	}
	double ChainElevator::Getcount()
	{
		//////////////////取得兩電梯移動得比值//////////////////////
		double scale = ((double)rangey2 / (double)rangey1);
		count = (rangey1 - abs(y - aimy1)) * scale;
		return count;
	}
	int ChainElevator::GetaimY1()
	{
		return aimy1;
	}
	int ChainElevator::GetX2()
	{
		return x2;
	}
	int ChainElevator::GetY2()
	{
		return y2;
	}
	void ChainElevator::LoadBitmap()
	{
		//////////////////////讀檔////////////////////////////
		image.LoadBitmap(elevator, RGB(255, 255, 255));
		Elevator2.LoadBitmap(elevator, RGB(255, 255, 255));
	}
	void ChainElevator::onShow()
	{
		//////////////////////顯示////////////////////////////
		FallWall::ShowBitmap();
		Elevator2.SetTopLeft(x2, y2+(int)Getcount());
		Elevator2.ShowBitmap();
	}
}
```

lever.h

```
#ifndef LEVER_H
#define LEVER_H
namespace game_framework
{
	class lever
	{
	public:
		lever();
		void SetXY(int _x,int _y);
		int GetX();
		int GetY();
		CAnimation Getimage();
		void LoadBitmap();
		void OnMove();
		void OnShow();
	private:
		int x, y;
		CAnimation image;
	};
}
#endif
```

lever.cpp

```
#include "stdafx.h"
#include "Resource.h"
#include <mmsystem.h>
#include <ddraw.h>
#include "audio.h"
#include "gamelib.h"
#include "lever.h"
```

```
namespace game_framework
{
	lever::lever()
	{
	}
	void lever::SetXY(int _x, int _y)
	{
		x = _x, y = _y;
	}
	int lever::GetX()
	{
		return x;
	}
	int lever::GetY()
	{
		return y;
	}
	CAnimation lever::Getimage()
	{
		return image;
	}
	void lever::OnMove()
	{
		image.OnMove();
	}
	void lever::OnShow()
	{
		image.SetTopLeft(x, y);
		image.OnShow();
	}
	void lever::LoadBitmap()
	{
		image.AddBitmap(LEVER, RGB(255, 255, 255));
		image.AddBitmap(LEVER2, RGB(255, 255, 255));
	}
}
```

StartMovie.h

```
#ifndef STARTMOVIE
#define STARTMOVIE
namespace game_framework {
	class Movie {
	public:
		Movie();
		void LoadBitmap();
		void SetIceXY(int _x,int _y);
		void SetFireXY(int _x, int _y);
		void Restart();
		void OnMove();
		void OnShow();
		bool Getend();
	private:
		CAnimation iceman;
		CAnimation fireman;
		int delaycount;
	};
}
#endif
```

StartMovie.cpp

```
#include "stdafx.h"
#include "Resource.h"
#include <mmsystem.h>
#include <ddraw.h>
#include "audio.h"
#include "gamelib.h"
#include "StartMovie.h"
namespace game_framework {
	Movie::Movie()
	{
		delaycount = 30;
	}
	void Movie::LoadBitmap()
	{
		//////////////////////////////////////讀取圖片//////////////////////////////////////
		iceman.AddBitmap(ICEMAN, RGB(255, 255, 255));
		iceman.AddBitmap(START1, RGB(0, 0, 0));
		iceman.AddBitmap(START2, RGB(0, 0, 0));
```

```
		iceman.AddBitmap(START3, RGB(0, 0, 0));
		fireman.AddBitmap(FIRE, RGB(255, 255, 255));
		fireman.AddBitmap(START1, RGB(0, 0, 0));
		fireman.AddBitmap(START2, RGB(0, 0, 0));
		fireman.AddBitmap(START3, RGB(0, 0, 0));
	}
	void Movie::SetIceXY(int _x, int _y)
	{
		iceman.SetTopLeft(_x, _y);
	}
	void Movie::SetFireXY(int _x, int _y)
	{
		fireman.SetTopLeft(_x, _y);
	}
	void Movie::Restart()
	{
		iceman.Reset();
		fireman.Reset();
		delaycount = 30;//開始總時間
	}
	void Movie::OnMove()
	{
		////////////////////////////////////圖片運行//////////////////////////////////////
		if (!iceman.IsFinalBitmap() && !fireman.IsFinalBitmap())
		{
			iceman.OnMove();
			fireman.OnMove();
		}
	}
	void Movie::OnShow()
	{
		iceman.OnShow();
		fireman.OnShow();
	}
	bool Movie::Getend()
	{
		if (iceman.IsFinalBitmap() && fireman.IsFinalBitmap() && delaycount--<=0) return true;//開頭畫面結束
		return false;
	}
}
```

Wind.h

```
#ifndef WIND_H
#define WIND_H
#include "man.h"
namespace game_framework {
	class Wind {
	public:
		Wind();
		void LoadBitmap();
		void Init();
		void Count();
		void onMove(man *player);
		void OnShow();
	private:
		int times;
		int direction;
		CMovingBitmap image1,image2,image3;
	};
}
#endif
```

Wind.cpp

```
#include "stdafx.h"
#include "Resource.h"
#include <mmsystem.h>
#include <ddraw.h>
#include "audio.h"
#include "gamelib.h"
#include <cstdlib>
#include <ctime>
#include "Wind.h"
namespace game_framework {
	Wind::Wind()
	{
		Init();
	}
	void Wind::LoadBitmap()
```

```
	{
		//////////////////////////讀取圖片//////////////////////////////////
		image1.LoadBitmap(WIND, RGB(255, 255, 255));
		image2.LoadBitmap(Wind0, RGB(255, 255, 255));
		image3.LoadBitmap(WIND2, RGB(255, 255, 255));
		image1.SetTopLeft(290, 20);
		image2.SetTopLeft(290, 20);
		image3.SetTopLeft(290, 20);
		//////////////////////////////////////////////////////////////////////
	}
	void Wind::Init()
	{
		unsigned seed;
		seed = (unsigned)time(NULL); // 取得亂數種子
		srand(seed); // 以時間序列當亂數種子
		times = 0;
	}
	void Wind::Count()
	{
		times++;
	}
	void Wind::onMove(man *player)
	{
		if (times % 900 == 0)
			direction = rand() % 3 - 1;//三種風向抽一種
		player->SetXY(player->GetX1() + direction, player->GetY1());
	}
	void Wind::OnShow()
	{
		if (direction == -1) image1.ShowBitmap();
		if (direction == 0) image2.ShowBitmap();
		if (direction == 1) image3.ShowBitmap();
	}
}
```

Fireballmovie.h

```
#ifndef FIREBALLMOVIE_H
#define FIREBALLMOVIE_H
namespace game_framework {
	class FireBallMovie {
	public:
		FireBallMovie();
		void Init();
		void LoadBitmap();
		void OnShow();
		void OnMove();
		void isEnd();
		bool isStart();
	private:
		bool Stop,Start;
		CMovingBitmap image;
		int delaycount,x,y;
	};
}
#endif
```

Fireballmovie.cpp

```
#include "stdafx.h"
#include "Resource.h"
#include <mmsystem.h>
#include <ddraw.h>
#include "audio.h"
#include "gamelib.h"
#include "Fireballmovie.h"
namespace game_framework {
	FireBallMovie::FireBallMovie()
	{
		Init();
		Start = false;
	}
	void FireBallMovie::Init()
	{
		x = 640;
		y = 100;
		Start = true;
		Stop = true;
	}
	//////////////////////////////////////////讀取圖片//////////////////////////////////////////
```

```
	void FireBallMovie::LoadBitmap()
	{
		image.LoadBitmap(FIREBALLMOVIE);
	}
	/////////////////////////////////////////顯示技能圖片/////////////////////////////////////
	void FireBallMovie::OnShow()
	{
		image.SetTopLeft(x, y);
		image.ShowBitmap();
		isEnd();
	}
	/////////////////////////////////////////移動技能圖片/////////////////////////////////////
	void FireBallMovie::OnMove()
	{
		if (x > 0) x -= 40;
		else if (x == 0 && Stop)
		{
			delaycount = 30;
			Stop = false;
			CAudio::Instance()->Play(7, false);
		}
		else if (delaycount-- <= 0) x -=40;
	}
	/////////////////////////////////////////判斷是否完成圖片移動效果/////////////////////////////
	void FireBallMovie::isEnd()
	{
		if (x == -640) Start = false;
	}
	//////////////////////////////////////////////////////////////////////////////////////////
	bool FireBallMovie::isStart()
	{
		return Start;
	}
}
```

Iceballmovie.h

```
#ifndef ICEBALLMOVIE_H
#define ICEBALLMOVIE_H
namespace game_framework {
	class IceBallMovie {
	public:
		IceBallMovie();
		void Init();
		void LoadBitmap();
		void OnShow();
		void OnMove();
		void isEnd();
		bool isStart();
	private:
		bool Stop, Start;
		CMovingBitmap image;
		int delaycount, x, y;
	};
}
#endif
```

Iceballmovie.cpp

```
#include "stdafx.h"
#include "Resource.h"
#include <mmsystem.h>
#include <ddraw.h>
#include "audio.h"
#include "gamelib.h"
#include "Iceballmovie.h"
namespace game_framework {
	IceBallMovie::IceBallMovie()
	{
		Init();
		Start = false;
	}
	void IceBallMovie::Init()
	{
		x = 640;
		y = 100;
		Start = true;
		Stop = true;
	}
	/////////////////////////////////////////讀取圖片/////////////////////////////////////
```

```
	void IceBallMovie::LoadBitmap()
	{
		image.LoadBitmap(ICEBALLMOVIE);
	}
	//////////////////////////////////////////顯示技能圖片//////////////////////////////////////
	void IceBallMovie::OnShow()
	{
		image.SetTopLeft(x, y);
		image.ShowBitmap();
		isEnd();
	}
	//////////////////////////////////////////移動技能圖片//////////////////////////////////////
	void IceBallMovie::OnMove()
	{
		if (x > 0) x -= 40;
		else if (x == 0 && Stop)
		{
			delaycount = 40;
			Stop = false;
			CAudio::Instance()->Play(8, false);
		}
		else if (delaycount-- <= 0) x -= 40;
	}
	//////////////////////////////////////////判斷是否完成圖片移動效果//////////////////////////////
	void IceBallMovie::isEnd()
	{
		if (x == -640) Start = false;
	}
	//////////////////////////////////////////////////////////////////////////////////////////
	bool IceBallMovie::isStart()
	{
		return Start;
	}
}
```

ingamemovie.h

```
#ifndef INGAMEMOVIE_H
#define INGAMEMOVIE_H
namespace game_framework {
	class ingamemovie {
	public:
		ingamemovie();
		void Init();
		void LoadBitmap();
		void OnShow();
		void OnMove();
		void SetUp();
		void SetDown();
		void GoMenu();
		void GoRetry();
		void GoInit();
		void ResetPos();
		bool isOK();
	private:
		bool up, down;
		CAnimation image;
		int	x, y;
	};
}
#endif
```

ingamemovie.cpp

```
#include "stdafx.h"
#include "Resource.h"
#include <mmsystem.h>
#include <ddraw.h>
#include "audio.h"
#include "gamelib.h"
#include "ingamemovie.h"
namespace game_framework {
	ingamemovie::ingamemovie()
	{
		x = 145;
		y = 460;

	}
	void ingamemovie::Init()
	{
```

```
		up = false;
		down = false;
	}
	//////////////////////////////////////////重置圖片//////////////////////////////////////////
	void ingamemovie::ResetPos()
	{
		x = 145;
		y = 460;
	}
	//////////////////////////////////////////////////////////////////////////////////////////
	//////////////////////////////////////////讀取圖片//////////////////////////////////////////
	void ingamemovie::LoadBitmap()
	{
		image.AddBitmap(ingame,RGB(0, 0, 0));
		image.AddBitmap(ingame_menu,RGB(0,0,0));
		image.AddBitmap(ingame_retry,RGB(0, 0, 0));
		image.SetDelayCount(1);
		image.Reset();
	}
	//////////////////////////////////////////////////////////////////////////////////////////
	//////////////////////////////////////////顯示圖片//////////////////////////////////////////
	void ingamemovie::OnShow()
	{
		image.SetTopLeft(x, y);
		image.OnShow();
	}
	//////////////////////////////////////////////////////////////////////////////////////////
	//////////////////////////////////////執行圖片上下滑動//////////////////////////////////////
	void ingamemovie::OnMove()
	{
		if (y > 80 && up) {
			y -= 40;
		}
		else if (y < 460 && down) {
			y += 20;
		}
		else if( x== 145 && y == 460 || x == 145 && y == 80)
		{
			Init();
		}
	}
	//////////////////////////////////////////////////////////////////////////////////////////
	//////////////////////////////////////圖片上下滑動切換//////////////////////////////////////
	void ingamemovie::SetUp()
	{
		up = true;
		down = false;
	}

	void ingamemovie::SetDown()
	{
		up = false;
		down = true;
	}
	//////////////////////////////////////////////////////////////////////////////////////////
	//////////////////////////////////////判斷圖片是否滑動到定位//////////////////////////////////
	bool ingamemovie::isOK()
	{
		if(x== 145 && y == 460) return true;
		else return false;

	}
	//////////////////////////////////////////////////////////////////////////////////////////
	void ingamemovie::GoMenu() {
		image.Movetonum(1);
	}
	void ingamemovie::GoRetry() {
		image.Movetonum(2);
	}
	void ingamemovie::GoInit() {
		image.Movetonum(0);
	}
}
```

Sasuke.h

```
#ifndef SASUKE_H
#define SASUKE_H
```

```
#include "SkillStart.h"
#include "SasukeFireball.h"
#include "Shurikan.h"
namespace game_framework {
	class Sasuke
	{
	public:
		Sasuke();
		~Sasuke();
		int		GetX1();					// 佐助左上角 x 座標
		int		GetY1();					// 佐助左上角 y 座標
		int		GetX2();					// 佐助右下角 x 座標
		int		GetY2();					// 佐助右下角 y 座標
		int		Gety();
		int		Getx();
		void Initialize();				// 設定佐助為初始值
		void LoadBitmap();				// 載入圖形
		void OnMove();
		void setfloor(int _floor);
		void jump();					// 跳躍
		void OnShow();						// 將佐助圖形貼到畫面
		void SetMovingLeft(bool flag);	// 設定是否正在往左移動
		void SetMovingRight(bool flag); // 設定是否正在往右移動
		void SetMovingUp(bool flag);	// 設定是否正在往上移動
		void SetAttacked(int flag);
		void SetDie();
		void SetDie(int Sub);
		void SetMove(bool flag);
		void SetisMolding(bool flag);
		bool GetMovingLeft();
		bool GetMovingRight();
		bool GetisDie();
		void SubBlood(int Sub);
		int GetDirection();
		int GetAttack();
		bool GetisSkill();
		bool GetisMolding();
		void SetXY(int nx, int ny);		// 設定佐助左上角座標
		void Addattacknum();
		bool GetMove();
		int Getblood();
		int GetMagic();
		int GetCurrentBitmapNumber();
		int GetSkillCurrentBitmapNumber();
		skillstart* Getskillstart();
		vector<SasukeFireball*> GetFireball();
		vector<Shurikan*> GetShurikan();
		void AttackMovie1();
		void AttackMovie2();
		void AttackMovie3();
		void AttackMovie4();
		int GetChakura();
		void ChangeChakura(int _chakura);
		void SetDirection(int _direction);
		void SetFireballDelete(int flag);
		void SetShurikanDelete(int flag);
		bool GetSkillsuccess();
		void SetSkillsuccess();
		CAnimation* GetSkillsuccessMovie();
		void RunSkillsuccessMovie();
		void Deleteobject();
		void Attack();
		void Attacked();
		void Molding();
		void ThrowMove();
		void MovePlayer();
		void Die();
		void Stand();
		void SetSkill();
	protected:
		int floor;
		int velocity;			// 目前的速度(點/次)
		int initial_velocity;	// 初始速度
		bool rising;				// true 表上升、false 表下降
		int x, y;					// 佐助左上角座標
		bool isMovingLeft;		// 是否正在往左移動
```

```
        bool isMovingRight;                 // 是否正在往右移動
        bool isMovingUp;                    // 是否正在往上移動
        bool isMolding;
        bool isSkillFail;
        bool isDie;
        int isAttacked;
        bool isShooting;
        bool Skillsuccess;
        int direction;
        int step_size;
        int magic;
        int Moldingdelay;
        bool Move;
        int Diedelay, chakuradelay, delay;
        CAnimation animation;               // 利用動畫作圖形
        CAnimation SkillMovie;
        CAnimation SkillsuccessMovie;
        skillstart Movie, Movie2;
        CMovingBitmap MagicBar;
        CMovingBitmap Black;
        vector<SasukeFireball*> Fireball;
        vector<Shurikan*> shurikan;
        int MagicX, MagicY;
        int Blood;
        vector <char> combo;
        bool Movieflag;
        int attacknum;
        bool isSkill;
        int chakura;
        int skill3direction,skill3delay, skill1delay;
        int damage;
    };
}
#endif
```

Sasuke.cpp

```
#include "stdafx.h"
#include "Resource.h"
#include <mmsystem.h>
#include <ddraw.h>
#include "audio.h"
#include "gamelib.h"
#include "Sasuke.h"


namespace game_framework {
    Sasuke::Sasuke()
    {
        unsigned seed;
        seed = (unsigned)time(NULL); // 取得時
        srand(seed); // 以時間序列當亂數種子
    }
    Sasuke::~Sasuke()
    {

    }
    int Sasuke::GetX1()
    {
        ////////////////////////////////////////////////////////圖片 x 座標修正//////////////////////////////////////////////
        if (direction == 0) return x;
        if (Skillsuccess) return x + (24 - SkillsuccessMovie.Width()) * 2;
        if (isSkill && direction == 1) return x + (24 - SkillMovie.Width()) * 2;
        return x + (24 - animation.Width())*2;
        ////////////////////////////////////////////////////////////////////////////////////////////////////////////////
    }
    int Sasuke::GetY1()
    {
        ////////////////////////////////////////////////////////圖片 y 座標修正//////////////////////////////////////////////
        if (Skillsuccess) return y + (36 - SkillsuccessMovie.Height()) * 2;
        if(isSkill) return y + (36 - SkillMovie.Height()) * 2;
        return y + (36 - animation.Height()) * 2;
        ////////////////////////////////////////////////////////////////////////////////////////////////////////////////
    }
    int Sasuke::GetX2()
    {
        if(Skillsuccess) return GetX1() + SkillsuccessMovie.Width() * 2;
        if(isSkill) return GetX1() + SkillMovie.Width() * 2;
```

```
	return GetX1() + animation.Width()*2;
}
int Sasuke::GetY2()
{
	if(Skillsuccess) return GetY1() + SkillsuccessMovie.Height() * 2;
	if (isSkill) return GetY1() + SkillMovie.Height() * 2;
	return GetY1() + animation.Height()*2;
}
int Sasuke::Gety()
{
	return y;
}
int Sasuke::Getx()
{
	return x;
}
void Sasuke::Initialize()
{
	//////////////////////////////////////////////////初始化//////////////////////////////////////////////////////////////
	floor = 460;
	const int X_POS = 100;
	const int Y_POS = 460 - 39;
	const int INITIAL_VELOCITY = 18;
	initial_velocity = INITIAL_VELOCITY;
	velocity = initial_velocity;
	x = X_POS;
	y = Y_POS;
	Skillsuccess = Movieflag= isSkill = isSkillFail = isMolding = isDie = isShooting = isAttacked = rising = isMovingLeft = isMovingRight =
	isMovingUp = false;
	step_size = 10;
	Move = true;
	skill3direction=attacknum = direction = magic = 0;
	Blood = 128;
	Diedelay = 30;
	delay = 2;
	chakura = 64;
	chakuradelay= skill1delay =8;
	skill3delay = 4;
	animation.Reset();
	SkillsuccessMovie.Reset();
	SkillMovie.Reset();
	damage = 1;
	//////////////////////////////////////////////////////////////////////////////////////////////////////////////////////
}
void Sasuke::LoadBitmap()
{
	//////////////////////////////////////////////////////////讀取圖片//////////////////////////////////////////
	animation.AddBitmap(SASUKESTAND1, RGB(73, 176, 255));	//0
	animation.AddBitmap(SASUKESTAND1, RGB(73, 176, 255));	//1
	animation.AddBitmap(SASUKESTAND1, RGB(73, 176, 255));	//2
	animation.AddBitmap(SASUKESTAND2, RGB(73, 176, 255));	//3
	animation.AddBitmap(SASUKESTAND2, RGB(73, 176, 255));		//4
	animation.AddBitmap(SASUKESTAND2, RGB(73, 176, 255));		//5
	animation.AddBitmap(SASUKESTAND3, RGB(73, 176, 255));		//6
	animation.AddBitmap(SASUKESTAND3, RGB(73, 176, 255));		//7
	animation.AddBitmap(SASUKESTAND3, RGB(73, 176, 255));		//8
	animation.AddBitmap(SASUKESTAND4, RGB(73, 176, 255));		//9
	animation.AddBitmap(SASUKESTAND4, RGB(73, 176, 255));		//10
	animation.AddBitmap(SASUKESTAND4, RGB(73, 176, 255));		//11
	animation.AddBitmap(SASUKERUN1, RGB(73, 176, 255));		//12
	animation.AddBitmap(SASUKERUN1, RGB(73, 176, 255));		//13
	animation.AddBitmap(SASUKERUN1, RGB(73, 176, 255));		//14
	animation.AddBitmap(SASUKERUN2, RGB(73, 176, 255));		//15
	animation.AddBitmap(SASUKERUN2, RGB(73, 176, 255));		//16
	animation.AddBitmap(SASUKERUN2, RGB(73, 176, 255));		//17
	animation.AddBitmap(SASUKEJUMP1, RGB(73, 176, 255));		//18
	animation.AddBitmap(SASUKEJUMP2, RGB(73, 176, 255));		//19
	animation.AddBitmap(SASUKEJUMPKICK1, RGB(73, 176, 255));	//20
	animation.AddBitmap(SASUKEJUMPKICK1, RGB(73, 176, 255));	//21
	animation.AddBitmap(SASUKEJUMPKICK1, RGB(73, 176, 255));	//22
	animation.AddBitmap(SASUKEJUMPKICK2, RGB(73, 176, 255));	//23
	animation.AddBitmap(SASUKEJUMPKICK2, RGB(73, 176, 255));	//24
	animation.AddBitmap(SASUKEJUMPKICK2, RGB(73, 176, 255));	//25
	animation.AddBitmap(SASUKEJUMPKICK3, RGB(73, 176, 255));	//26
	animation.AddBitmap(SASUKEJUMPKICK3, RGB(73, 176, 255));	//27
	animation.AddBitmap(SASUKEJUMPKICK3, RGB(73, 176, 255));	//28
```

```
animation.AddBitmap(SASUKEDOWN1, RGB(73, 176, 255));          //29
animation.AddBitmap(SASUKEDOWN1, RGB(73, 176, 255));          //30
animation.AddBitmap(SASUKEDOWN1, RGB(73, 176, 255));          //31
animation.AddBitmap(SASUKECOMBO1, RGB(73, 176, 255));         //32
animation.AddBitmap(SASUKECOMBO1, RGB(73, 176, 255));         //33
animation.AddBitmap(SASUKECOMBO1, RGB(73, 176, 255));         //34
animation.AddBitmap(SASUKECOMBO2, RGB(73, 176, 255));         //35
animation.AddBitmap(SASUKECOMBO2, RGB(73, 176, 255));         //36
animation.AddBitmap(SASUKECOMBO2, RGB(73, 176, 255));         //37
animation.AddBitmap(SASUKECOMBO3, RGB(73, 176, 255));         //38
animation.AddBitmap(SASUKECOMBO3, RGB(73, 176, 255));         //39
animation.AddBitmap(SASUKECOMBO3, RGB(73, 176, 255));         //40
animation.AddBitmap(SASUKECOMBO4, RGB(73, 176, 255));         //41
animation.AddBitmap(SASUKECOMBO4, RGB(73, 176, 255));         //42
animation.AddBitmap(SASUKECOMBO4, RGB(73, 176, 255));         //43
animation.AddBitmap(SASUKECOMBO5, RGB(73, 176, 255));         //44
animation.AddBitmap(SASUKECOMBO5, RGB(73, 176, 255));         //45
animation.AddBitmap(SASUKECOMBO5, RGB(73, 176, 255));         //46
animation.AddBitmap(SASUKECOMBO6, RGB(73, 176, 255));         //47
animation.AddBitmap(SASUKECOMBO6, RGB(73, 176, 255));         //48
animation.AddBitmap(SASUKECOMBO6, RGB(73, 176, 255));         //49
animation.AddBitmap(SASUKECOMBO7, RGB(73, 176, 255));         //50
animation.AddBitmap(SASUKECOMBO7, RGB(73, 176, 255));         //51
animation.AddBitmap(SASUKECOMBO7, RGB(73, 176, 255));         //52
animation.AddBitmap(SASUKECOMBO8, RGB(73, 176, 255));         //53
animation.AddBitmap(SASUKECOMBO8, RGB(73, 176, 255));         //54
animation.AddBitmap(SASUKECOMBO8, RGB(73, 176, 255));         //55
animation.AddBitmap(SASUKECOMBO9, RGB(73, 176, 255));         //56
animation.AddBitmap(SASUKECOMBO9, RGB(73, 176, 255));         //57
animation.AddBitmap(SASUKECOMBO9, RGB(73, 176, 255));         //58
animation.AddBitmap(SASUKECOMBO10, RGB(73, 176, 255)); //59
animation.AddBitmap(SASUKECOMBO10, RGB(73, 176, 255)); //60
animation.AddBitmap(SASUKECOMBO10, RGB(73, 176, 255)); //61
animation.AddBitmap(SASUKEDIE1, RGB(73, 176, 255));           //62
animation.AddBitmap(SASUKEDIE1, RGB(73, 176, 255));           //63
animation.AddBitmap(SASUKEDIE1, RGB(73, 176, 255));           //64
animation.AddBitmap(SASUKEDIE1, RGB(73, 176, 255));           //65
animation.AddBitmap(SASUKEDIE1, RGB(73, 176, 255));           //66
animation.AddBitmap(SASUKEDIE1, RGB(73, 176, 255));           //67
animation.AddBitmap(SASUKEDIE2, RGB(73, 176, 255));           //68
animation.AddBitmap(SASUKEDIE2, RGB(73, 176, 255));           //69
animation.AddBitmap(SASUKEDIE2, RGB(73, 176, 255));           //70
animation.AddBitmap(SASUKEDIE2, RGB(73, 176, 255));           //71
animation.AddBitmap(SASUKEDIE2, RGB(73, 176, 255));           //72
animation.AddBitmap(SASUKEDIE2, RGB(73, 176, 255));           //73
animation.AddBitmap(SASUKEDIE3, RGB(73, 176, 255));           //74
animation.AddBitmap(SASUKEDIE3, RGB(73, 176, 255));           //75
animation.AddBitmap(SASUKEDIE3, RGB(73, 176, 255));           //76
animation.AddBitmap(SASUKEDIE3, RGB(73, 176, 255));           //77
animation.AddBitmap(SASUKEDIE3, RGB(73, 176, 255));           //78
animation.AddBitmap(SASUKEDIE3, RGB(73, 176, 255));           //79
animation.AddBitmap(SASUKEDIE4, RGB(73, 176, 255));           //80
animation.AddBitmap(SASUKEDIE4, RGB(73, 176, 255));           //81
animation.AddBitmap(SASUKEDIE5, RGB(73, 176, 255));           //82
animation.AddBitmap(SASUKEDIE5, RGB(73, 176, 255));           //83
animation.AddBitmap(SASUKEDIE6, RGB(73, 176, 255));           //84
animation.AddBitmap(SASUKEDIE6, RGB(73, 176, 255));           //85
animation.AddBitmap(SASUKEDIE7, RGB(73, 176, 255));           //86
animation.AddBitmap(SASUKEDIE7, RGB(73, 176, 255));           //87
animation.AddBitmap(SASUKEDIE8, RGB(73, 176, 255));           //88
animation.AddBitmap(SASUKEDIE8, RGB(73, 176, 255));           //89
animation.AddBitmap(SASUKEDIE8, RGB(73, 176, 255));           //90
animation.AddBitmap(SASUKEDIE8, RGB(73, 176, 255));           //91
animation.AddBitmap(SASUKEDIE8, RGB(73, 176, 255));           //92
animation.AddBitmap(SASUKEDIE8, RGB(73, 176, 255));           //93
animation.AddBitmap(SASUKEDIE9, RGB(73, 176, 255));           //94
animation.AddBitmap(SASUKEDIE9, RGB(73, 176, 255));           //95
animation.AddBitmap(transe, RGB(255, 255, 255));          //96
animation.AddBitmap(transe, RGB(255, 255, 255));          //97
animation.AddBitmap(SASUKEDIE9, RGB(73, 176, 255));           //98
animation.AddBitmap(SASUKEDIE9, RGB(73, 176, 255));           //99
animation.AddBitmap(transe, RGB(255, 255, 255));          //100
animation.AddBitmap(transe, RGB(255, 255, 255));          //101
animation.AddBitmap(SASUKEDIE9, RGB(73, 176, 255));           //102
animation.AddBitmap(SASUKEDIE9, RGB(73, 176, 255));           //103
animation.AddBitmap(transe, RGB(255, 255, 255));          //104
```

```
animation.AddBitmap(transe, RGB(255, 255, 255));          //105
animation.AddBitmap(SASUKEDIE9, RGB(73, 176, 255));                //106
animation.AddBitmap(SASUKEDIE9, RGB(73, 176, 255));                //107
animation.AddBitmap(transe, RGB(255, 255, 255));          //108
animation.AddBitmap(transe, RGB(255, 255, 255));          //109
animation.AddBitmap(SASUKEDIE9, RGB(73, 176, 255));                //110
animation.AddBitmap(SASUKEHURT1, RGB(73, 176, 255));               //111
animation.AddBitmap(SASUKEHURT1, RGB(73, 176, 255));               //112
animation.AddBitmap(SASUKEHURT1, RGB(73, 176, 255));               //113
animation.AddBitmap(SASUKEMOLDING1, RGB(73, 176, 255));        //114
animation.AddBitmap(SASUKEMOLDING1, RGB(73, 176, 255));        //115
animation.AddBitmap(SASUKEMOLDING1, RGB(73, 176, 255));        //116
animation.AddBitmap(SASUKEMOLDING2, RGB(73, 176, 255));        //117
animation.AddBitmap(SASUKEMOLDING2, RGB(73, 176, 255));        //118
animation.AddBitmap(SASUKEMOLDING2, RGB(73, 176, 255));        //119
animation.AddBitmap(SASUKEMOLDING3, RGB(73, 176, 255));        //120
animation.AddBitmap(SASUKEMOLDING3, RGB(73, 176, 255));        //121
animation.AddBitmap(SASUKEMOLDING3, RGB(73, 176, 255));        //122
animation.AddBitmap(SASUKEMOLDING4, RGB(73, 176, 255));        //123
animation.AddBitmap(SASUKEMOLDING4, RGB(73, 176, 255));        //124
animation.AddBitmap(SASUKEMOLDING4, RGB(73, 176, 255));        //125
animation.AddBitmap(SASUKEMOLDING4, RGB(73, 176, 255));        //126
animation.AddBitmap(SASUKEMOLDING5, RGB(73, 176, 255));        //127
animation.AddBitmap(SASUKEMOLDING5, RGB(73, 176, 255));        //128
animation.AddBitmap(SASUKEMOLDING5, RGB(73, 176, 255));        //129
animation.AddBitmap(SASUKEMOLDING5, RGB(73, 176, 255));        //130
animation.AddBitmap(SASUKESKILLFAIL1, RGB(73, 176, 255));      //131
animation.AddBitmap(SASUKESKILLFAIL1, RGB(73, 176, 255));      //132
animation.AddBitmap(SASUKESKILLFAIL1, RGB(73, 176, 255));      //133
animation.AddBitmap(SASUKESKILLFAIL2, RGB(73, 176, 255));      //134
animation.AddBitmap(SASUKESKILLFAIL2, RGB(73, 176, 255));      //135
animation.AddBitmap(SASUKESKILLFAIL2, RGB(73, 176, 255));      //136
animation.AddBitmap(SASUKESKILLFAIL3, RGB(73, 176, 255));      //137
animation.AddBitmap(SASUKESKILLFAIL3, RGB(73, 176, 255));      //138
animation.AddBitmap(SASUKESKILLFAIL3, RGB(73, 176, 255));      //139
animation.AddBitmap(SASUKESTAND5, RGB(73, 176, 255));              //140
animation.AddBitmap(SASUKESTAND5, RGB(73, 176, 255));              //141
animation.AddBitmap(SASUKESTAND5, RGB(73, 176, 255));              //142
animation.AddBitmap(SASUKESTAND6, RGB(73, 176, 255));              //143
animation.AddBitmap(SASUKESTAND6, RGB(73, 176, 255));              //144
animation.AddBitmap(SASUKESTAND6, RGB(73, 176, 255));              //145
animation.AddBitmap(SASUKESTAND7, RGB(73, 176, 255));              //146
animation.AddBitmap(SASUKESTAND7, RGB(73, 176, 255));              //147
animation.AddBitmap(SASUKESTAND7, RGB(73, 176, 255));              //148
animation.AddBitmap(SASUKESTAND8, RGB(73, 176, 255));              //149
animation.AddBitmap(SASUKESTAND8, RGB(73, 176, 255));              //150
animation.AddBitmap(SASUKESTAND8, RGB(73, 176, 255));              //151
animation.AddBitmap(SASUKERUN3, RGB(73, 176, 255));                //152
animation.AddBitmap(SASUKERUN3, RGB(73, 176, 255));                //153
animation.AddBitmap(SASUKERUN3, RGB(73, 176, 255));                //154
animation.AddBitmap(SASUKERUN4, RGB(73, 176, 255));                //155
animation.AddBitmap(SASUKERUN4, RGB(73, 176, 255));                //156
animation.AddBitmap(SASUKERUN4, RGB(73, 176, 255));                //157
animation.AddBitmap(SASUKEJUMP3, RGB(73, 176, 255));               //158
animation.AddBitmap(SASUKEJUMP4, RGB(73, 176, 255));               //159
animation.AddBitmap(SASUKEJUMPKICK4, RGB(73, 176, 255));       //160
animation.AddBitmap(SASUKEJUMPKICK4, RGB(73, 176, 255));       //161
animation.AddBitmap(SASUKEJUMPKICK4, RGB(73, 176, 255));       //162
animation.AddBitmap(SASUKEJUMPKICK5, RGB(73, 176, 255));       //163
animation.AddBitmap(SASUKEJUMPKICK5, RGB(73, 176, 255));       //164
animation.AddBitmap(SASUKEJUMPKICK5, RGB(73, 176, 255));       //165
animation.AddBitmap(SASUKEJUMPKICK6, RGB(73, 176, 255));       //166
animation.AddBitmap(SASUKEJUMPKICK6, RGB(73, 176, 255));       //167
animation.AddBitmap(SASUKEJUMPKICK6, RGB(73, 176, 255));       //168
animation.AddBitmap(SASUKEDOWN2, RGB(73, 176, 255));               //169
animation.AddBitmap(SASUKEDOWN2, RGB(73, 176, 255));               //170
animation.AddBitmap(SASUKEDOWN2, RGB(73, 176, 255));               //171
animation.AddBitmap(SASUKECOMBO11, RGB(73, 176, 255)); //172
animation.AddBitmap(SASUKECOMBO11, RGB(73, 176, 255)); //173
animation.AddBitmap(SASUKECOMBO11, RGB(73, 176, 255)); //174
animation.AddBitmap(SASUKECOMBO12, RGB(73, 176, 255)); //175
animation.AddBitmap(SASUKECOMBO12, RGB(73, 176, 255)); //176
animation.AddBitmap(SASUKECOMBO12, RGB(73, 176, 255)); //177
animation.AddBitmap(SASUKECOMBO13, RGB(73, 176, 255)); //178
animation.AddBitmap(SASUKECOMBO13, RGB(73, 176, 255)); //179
animation.AddBitmap(SASUKECOMBO13, RGB(73, 176, 255)); //180
```

```
animation.AddBitmap(SASUKECOMBO14, RGB(73, 176, 255)); //181
animation.AddBitmap(SASUKECOMBO14, RGB(73, 176, 255)); //182
animation.AddBitmap(SASUKECOMBO14, RGB(73, 176, 255)); //183
animation.AddBitmap(SASUKECOMBO15, RGB(73, 176, 255)); //184
animation.AddBitmap(SASUKECOMBO15, RGB(73, 176, 255)); //185
animation.AddBitmap(SASUKECOMBO15, RGB(73, 176, 255)); //186
animation.AddBitmap(SASUKECOMBO16, RGB(73, 176, 255)); //187
animation.AddBitmap(SASUKECOMBO16, RGB(73, 176, 255)); //188
animation.AddBitmap(SASUKECOMBO16, RGB(73, 176, 255)); //189
animation.AddBitmap(SASUKECOMBO17, RGB(73, 176, 255)); //190
animation.AddBitmap(SASUKECOMBO17, RGB(73, 176, 255)); //191
animation.AddBitmap(SASUKECOMBO17, RGB(73, 176, 255)); //192
animation.AddBitmap(SASUKECOMBO18, RGB(73, 176, 255)); //193
animation.AddBitmap(SASUKECOMBO18, RGB(73, 176, 255)); //194
animation.AddBitmap(SASUKECOMBO18, RGB(73, 176, 255)); //195
animation.AddBitmap(SASUKECOMBO19, RGB(73, 176, 255)); //196
animation.AddBitmap(SASUKECOMBO19, RGB(73, 176, 255)); //197
animation.AddBitmap(SASUKECOMBO19, RGB(73, 176, 255)); //198
animation.AddBitmap(SASUKECOMBO20, RGB(73, 176, 255)); //199
animation.AddBitmap(SASUKECOMBO20, RGB(73, 176, 255)); //200
animation.AddBitmap(SASUKECOMBO20, RGB(73, 176, 255)); //201
animation.AddBitmap(SASUKEDIE10, RGB(73, 176, 255));          //202
animation.AddBitmap(SASUKEDIE10, RGB(73, 176, 255));          //203
animation.AddBitmap(SASUKEDIE10, RGB(73, 176, 255));          //204
animation.AddBitmap(SASUKEDIE10, RGB(73, 176, 255));          //205
animation.AddBitmap(SASUKEDIE10, RGB(73, 176, 255));          //206
animation.AddBitmap(SASUKEDIE10, RGB(73, 176, 255));          //207
animation.AddBitmap(SASUKEDIE11, RGB(73, 176, 255));          //208
animation.AddBitmap(SASUKEDIE11, RGB(73, 176, 255));          //209
animation.AddBitmap(SASUKEDIE11, RGB(73, 176, 255));          //210
animation.AddBitmap(SASUKEDIE11, RGB(73, 176, 255));          //211
animation.AddBitmap(SASUKEDIE11, RGB(73, 176, 255));          //212
animation.AddBitmap(SASUKEDIE11, RGB(73, 176, 255));          //213
animation.AddBitmap(SASUKEDIE12, RGB(73, 176, 255));          //214
animation.AddBitmap(SASUKEDIE12, RGB(73, 176, 255));          //215
animation.AddBitmap(SASUKEDIE12, RGB(73, 176, 255));          //216
animation.AddBitmap(SASUKEDIE12, RGB(73, 176, 255));          //217
animation.AddBitmap(SASUKEDIE12, RGB(73, 176, 255));          //218
animation.AddBitmap(SASUKEDIE12, RGB(73, 176, 255));          //219
animation.AddBitmap(SASUKEDIE13, RGB(73, 176, 255));          //220
animation.AddBitmap(SASUKEDIE13, RGB(73, 176, 255));          //221
animation.AddBitmap(SASUKEDIE14, RGB(73, 176, 255));          //222
animation.AddBitmap(SASUKEDIE14, RGB(73, 176, 255));          //223
animation.AddBitmap(SASUKEDIE15, RGB(73, 176, 255));          //224
animation.AddBitmap(SASUKEDIE15, RGB(73, 176, 255));          //225
animation.AddBitmap(SASUKEDIE16, RGB(73, 176, 255));          //226
animation.AddBitmap(SASUKEDIE16, RGB(73, 176, 255));          //227
animation.AddBitmap(SASUKEDIE17, RGB(73, 176, 255));          //228
animation.AddBitmap(SASUKEDIE17, RGB(73, 176, 255));          //229
animation.AddBitmap(SASUKEDIE17, RGB(73, 176, 255));          //230
animation.AddBitmap(SASUKEDIE17, RGB(73, 176, 255));          //231
animation.AddBitmap(SASUKEDIE17, RGB(73, 176, 255));          //232
animation.AddBitmap(SASUKEDIE17, RGB(73, 176, 255));          //233
animation.AddBitmap(SASUKEDIE18, RGB(73, 176, 255));          //234
animation.AddBitmap(SASUKEDIE18, RGB(73, 176, 255));          //235
animation.AddBitmap(transe, RGB(255, 255, 255));          //236
animation.AddBitmap(transe, RGB(255, 255, 255));          //237
animation.AddBitmap(SASUKEDIE18, RGB(73, 176, 255));          //238
animation.AddBitmap(SASUKEDIE18, RGB(73, 176, 255));          //239
animation.AddBitmap(transe, RGB(255, 255, 255));          //240
animation.AddBitmap(transe, RGB(255, 255, 255));          //241
animation.AddBitmap(SASUKEDIE18, RGB(73, 176, 255));          //242
animation.AddBitmap(SASUKEDIE18, RGB(73, 176, 255));          //243
animation.AddBitmap(transe, RGB(255, 255, 255));          //244
animation.AddBitmap(transe, RGB(255, 255, 255));          //245
animation.AddBitmap(SASUKEDIE18, RGB(73, 176, 255));          //246
animation.AddBitmap(SASUKEDIE18, RGB(73, 176, 255));          //247
animation.AddBitmap(transe, RGB(255, 255, 255));          //248
animation.AddBitmap(transe, RGB(255, 255, 255));          //249
animation.AddBitmap(SASUKEDIE18, RGB(73, 176, 255));          //250
animation.AddBitmap(SASUKEHURT2, RGB(73, 176, 255));          //251
animation.AddBitmap(SASUKEHURT2, RGB(73, 176, 255));          //252
animation.AddBitmap(SASUKEHURT2, RGB(73, 176, 255));          //253
animation.AddBitmap(SASUKEMOLDING7, RGB(73, 176, 255));       //254
animation.AddBitmap(SASUKEMOLDING7, RGB(73, 176, 255));       //255
animation.AddBitmap(SASUKEMOLDING8, RGB(73, 176, 255));       //256
```

```
animation.AddBitmap(SASUKEMOLDING8, RGB(73, 176, 255));	//257
animation.AddBitmap(SASUKEMOLDING8, RGB(73, 176, 255));	//258
animation.AddBitmap(SASUKEMOLDING8, RGB(73, 176, 255));	//259
animation.AddBitmap(SASUKEMOLDING9, RGB(73, 176, 255));	//260
animation.AddBitmap(SASUKEMOLDING9, RGB(73, 176, 255));	//261
animation.AddBitmap(SASUKEMOLDING9, RGB(73, 176, 255));	//262
animation.AddBitmap(SASUKEMOLDING10, RGB(73, 176, 255));	//263
animation.AddBitmap(SASUKEMOLDING10, RGB(73, 176, 255));	//264
animation.AddBitmap(SASUKEMOLDING10, RGB(73, 176, 255))	;//265
animation.AddBitmap(SASUKEMOLDING10, RGB(73, 176, 255));	//266
animation.AddBitmap(SASUKEMOLDING11, RGB(73, 176, 255));	//267
animation.AddBitmap(SASUKEMOLDING11, RGB(73, 176, 255));	//268
animation.AddBitmap(SASUKEMOLDING11, RGB(73, 176, 255));	//269
animation.AddBitmap(SASUKEMOLDING11, RGB(73, 176, 255));	//270
animation.AddBitmap(SASUKESKILLFAIL4, RGB(73, 176, 255));	//271
animation.AddBitmap(SASUKESKILLFAIL4, RGB(73, 176, 255));	//272
animation.AddBitmap(SASUKESKILLFAIL4, RGB(73, 176, 255));	//273
animation.AddBitmap(SASUKESKILLFAIL5, RGB(73, 176, 255));	//274
animation.AddBitmap(SASUKESKILLFAIL5, RGB(73, 176, 255));	//275
animation.AddBitmap(SASUKESKILLFAIL5, RGB(73, 176, 255));	//276
animation.AddBitmap(SASUKESKILLFAIL6, RGB(73, 176, 255));	//277
animation.AddBitmap(SASUKESKILLFAIL6, RGB(73, 176, 255));	//278
animation.AddBitmap(SASUKESKILLFAIL6, RGB(73, 176, 255));	//279
SkillMovie.AddBitmap(SASUKEHOUSENKA1, RGB(73, 176, 255));	//0
SkillMovie.AddBitmap(SASUKEHOUSENKA1, RGB(73, 176, 255));	//1
SkillMovie.AddBitmap(SASUKEHOUSENKA1, RGB(73, 176, 255));	//2
SkillMovie.AddBitmap(SASUKEHOUSENKA2, RGB(73, 176, 255));	//3
SkillMovie.AddBitmap(SASUKEHOUSENKA2, RGB(73, 176, 255));	//4
SkillMovie.AddBitmap(SASUKEHOUSENKA2, RGB(73, 176, 255));	//5
SkillMovie.AddBitmap(SASUKEHOUSENKA3, RGB(73, 176, 255));	//6
SkillMovie.AddBitmap(SASUKEHOUSENKA3, RGB(73, 176, 255));	//7
SkillMovie.AddBitmap(SASUKEHOUSENKA3, RGB(73, 176, 255));	//8
SkillMovie.AddBitmap(SASUKEHOUSENKA4, RGB(73, 176, 255));	//9
SkillMovie.AddBitmap(SASUKEHOUSENKA4, RGB(73, 176, 255));	//10
SkillMovie.AddBitmap(SASUKEHOUSENKA4, RGB(73, 176, 255));	//1
SkillMovie.AddBitmap(SASUKEHOUSENKA5, RGB(73, 176, 255));	//2
SkillMovie.AddBitmap(SASUKEHOUSENKA5, RGB(73, 176, 255));	//3
SkillMovie.AddBitmap(SASUKEHOUSENKA5, RGB(73, 176, 255));	//4
SkillMovie.AddBitmap(SASUKEHOUSENKA6, RGB(73, 176, 255));	//5
SkillMovie.AddBitmap(SASUKEHOUSENKA6, RGB(73, 176, 255));	//6
SkillMovie.AddBitmap(SASUKEHOUSENKA6, RGB(73, 176, 255));	//7
SkillMovie.AddBitmap(SASUKEHOUSENKA7, RGB(73, 176, 255));	//8
SkillMovie.AddBitmap(SASUKEHOUSENKA7, RGB(73, 176, 255));	//9
SkillMovie.AddBitmap(SASUKEHOUSENKA7, RGB(73, 176, 255));	//20
SkillMovie.AddBitmap(SASUKEHOUSENKA8, RGB(73, 176, 255));	//1
SkillMovie.AddBitmap(SASUKEHOUSENKA8, RGB(73, 176, 255));	//2
SkillMovie.AddBitmap(SASUKEHOUSENKA8, RGB(73, 176, 255));	//3
SkillMovie.AddBitmap(SASUKEHOUSENKA9, RGB(73, 176, 255));	//4
SkillMovie.AddBitmap(SASUKEHOUSENKA9, RGB(73, 176, 255));	//5
SkillMovie.AddBitmap(SASUKEHOUSENKA9, RGB(73, 176, 255));	//6
SkillMovie.AddBitmap(SASUKEHOUSENKA10, RGB(73, 176, 255));	//7
SkillMovie.AddBitmap(SASUKEHOUSENKA10, RGB(73, 176, 255));	//8
SkillMovie.AddBitmap(SASUKEHOUSENKA10, RGB(73, 176, 255));	//9
SkillMovie.AddBitmap(SASUKEHOUSENKA11, RGB(73, 176, 255));	//30
SkillMovie.AddBitmap(SASUKEHOUSENKA11, RGB(73, 176, 255));	//1
SkillMovie.AddBitmap(SASUKEHOUSENKA11, RGB(73, 176, 255));	//2
SkillMovie.AddBitmap(SASUKEHOUSENKA12, RGB(73, 176, 255));	//3
SkillMovie.AddBitmap(SASUKEHOUSENKA12, RGB(73, 176, 255));	//4
SkillMovie.AddBitmap(SASUKEHOUSENKA12, RGB(73, 176, 255));	//5
SkillMovie.AddBitmap(SASUKEHOUSENKA13, RGB(73, 176, 255));	//6
SkillMovie.AddBitmap(SASUKEHOUSENKA13, RGB(73, 176, 255));	//7
SkillMovie.AddBitmap(SASUKEHOUSENKA13, RGB(73, 176, 255));	//8
SkillMovie.AddBitmap(SASUKEHOUSENKA14, RGB(73, 176, 255));	//9
SkillMovie.AddBitmap(SASUKEHOUSENKA14, RGB(73, 176, 255));	//40
SkillMovie.AddBitmap(SASUKEHOUSENKA14, RGB(73, 176, 255));	//1
SkillMovie.AddBitmap(SASUKEHOUSENKA15, RGB(73, 176, 255));	//2
SkillMovie.AddBitmap(SASUKEHOUSENKA15, RGB(73, 176, 255));	//3
SkillMovie.AddBitmap(SASUKEHOUSENKA15, RGB(73, 176, 255));	//4
SkillMovie.AddBitmap(SASUKEHOUSENKA16, RGB(73, 176, 255));	//5
SkillMovie.AddBitmap(SASUKEHOUSENKA16, RGB(73, 176, 255));	//6
SkillMovie.AddBitmap(SASUKEHOUSENKA16, RGB(73, 176, 255));	//7
SkillMovie.AddBitmap(SASUKEHOUSENKA17, RGB(73, 176, 255));	//8
SkillMovie.AddBitmap(SASUKEHOUSENKA17, RGB(73, 176, 255));	//9
SkillMovie.AddBitmap(SASUKEHOUSENKA17, RGB(73, 176, 255));	//50
SkillMovie.AddBitmap(SASUKEHOUSENKA18, RGB(73, 176, 255));	//1
SkillMovie.AddBitmap(SASUKEHOUSENKA18, RGB(73, 176, 255));	//2
```

```
SkillMovie.AddBitmap(SASUKEHOUSENKA18, RGB(73, 176, 255));   //3
SkillMovie.AddBitmap(SASUKEHOUSENKA19, RGB(73, 176, 255));   //4
SkillMovie.AddBitmap(SASUKEHOUSENKA19, RGB(73, 176, 255));   //5
SkillMovie.AddBitmap(SASUKEHOUSENKA19, RGB(73, 176, 255));   //6
SkillMovie.AddBitmap(SASUKEHOUSENKA20, RGB(73, 176, 255));   //7
SkillMovie.AddBitmap(SASUKEHOUSENKA20, RGB(73, 176, 255));   //8
SkillMovie.AddBitmap(SASUKEHOUSENKA20, RGB(73, 176, 255));   //9
SkillMovie.AddBitmap(SASUKEHOUSENKA21, RGB(73, 176, 255));   //60
SkillMovie.AddBitmap(SASUKEHOUSENKA21, RGB(73, 176, 255));   //1
SkillMovie.AddBitmap(SASUKEHOUSENKA21, RGB(73, 176, 255));   //2
SkillMovie.AddBitmap(SASUKEHOUSENKA22, RGB(73, 176, 255));   //3
SkillMovie.AddBitmap(SASUKEHOUSENKA22, RGB(73, 176, 255));   //4
SkillMovie.AddBitmap(SASUKEHOUSENKA22, RGB(73, 176, 255));   //5
SkillMovie.AddBitmap(SASUKEHOUSENKA23, RGB(73, 176, 255));   //6
SkillMovie.AddBitmap(SASUKEHOUSENKA23, RGB(73, 176, 255));   //7
SkillMovie.AddBitmap(SASUKEHOUSENKA23, RGB(73, 176, 255));   //8
SkillMovie.AddBitmap(SASUKEHOUSENKA24, RGB(73, 176, 255));   //9
SkillMovie.AddBitmap(SASUKEHOUSENKA24, RGB(73, 176, 255));   //70
SkillMovie.AddBitmap(SASUKEHOUSENKA24, RGB(73, 176, 255));   //1
SkillMovie.AddBitmap(SASUKEHOUSENKA25, RGB(73, 176, 255));   //2
SkillMovie.AddBitmap(SASUKEHOUSENKA25, RGB(73, 176, 255));   //3
SkillMovie.AddBitmap(SASUKEHOUSENKA25, RGB(73, 176, 255));   //4
SkillMovie.AddBitmap(SASUKEHOUSENKA26, RGB(73, 176, 255));   //5
SkillMovie.AddBitmap(SASUKEHOUSENKA26, RGB(73, 176, 255));   //6
SkillMovie.AddBitmap(SASUKEHOUSENKA26, RGB(73, 176, 255));   //7
SkillMovie.AddBitmap(SASUKEHOUSENKA27, RGB(73, 176, 255));   //8
SkillMovie.AddBitmap(SASUKEHOUSENKA27, RGB(73, 176, 255));   //9
SkillMovie.AddBitmap(SASUKEHOUSENKA27, RGB(73, 176, 255));   //80
SkillMovie.AddBitmap(SASUKEHOUSENKA28, RGB(73, 176, 255));   //1
SkillMovie.AddBitmap(SASUKEHOUSENKA28, RGB(73, 176, 255));   //2
SkillMovie.AddBitmap(SASUKEHOUSENKA28, RGB(73, 176, 255));   //3
SkillMovie.AddBitmap(SASUKEHOUSENKA29, RGB(73, 176, 255));   //4
SkillMovie.AddBitmap(SASUKEHOUSENKA29, RGB(73, 176, 255));   //5
SkillMovie.AddBitmap(SASUKEHOUSENKA29, RGB(73, 176, 255));   //6
SkillMovie.AddBitmap(SASUKEHOUSENKA30, RGB(73, 176, 255));   //7
SkillMovie.AddBitmap(SASUKEHOUSENKA30, RGB(73, 176, 255));   //8
SkillMovie.AddBitmap(SASUKEHOUSENKA30, RGB(73, 176, 255));   //9
SkillMovie.AddBitmap(SASUKESKILL1, RGB(73, 176, 255));       //90
SkillMovie.AddBitmap(SASUKESKILL1, RGB(73, 176, 255));       //1
SkillMovie.AddBitmap(SASUKESKILL1, RGB(73, 176, 255));       //2
SkillMovie.AddBitmap(SASUKESKILL2, RGB(73, 176, 255));       //3
SkillMovie.AddBitmap(SASUKESKILL2, RGB(73, 176, 255));       //4
SkillMovie.AddBitmap(SASUKESKILL2, RGB(73, 176, 255));       //5
SkillMovie.AddBitmap(SASUKESKILL3, RGB(73, 176, 255));       //6
SkillMovie.AddBitmap(SASUKESKILL3, RGB(73, 176, 255));       //7
SkillMovie.AddBitmap(SASUKESKILL3, RGB(73, 176, 255));       //8
SkillMovie.AddBitmap(SASUKESKILL4, RGB(73, 176, 255));       //9
SkillMovie.AddBitmap(SASUKESKILL4, RGB(73, 176, 255));       //100
SkillMovie.AddBitmap(SASUKESKILL4, RGB(73, 176, 255));       //1
SkillMovie.AddBitmap(SASUKESKILL5, RGB(73, 176, 255));       //2
SkillMovie.AddBitmap(SASUKESKILL5, RGB(73, 176, 255));       //3
SkillMovie.AddBitmap(SASUKESKILL5, RGB(73, 176, 255));       //4
SkillMovie.AddBitmap(SASUKESKILL6, RGB(73, 176, 255));       //5
SkillMovie.AddBitmap(SASUKESKILL6, RGB(73, 176, 255));       //6
SkillMovie.AddBitmap(SASUKESKILL6, RGB(73, 176, 255));       //7
SkillMovie.AddBitmap(SASUKESKILL7, RGB(73, 176, 255));       //8
SkillMovie.AddBitmap(SASUKESKILL7, RGB(73, 176, 255));       //9
SkillMovie.AddBitmap(SASUKESKILL7, RGB(73, 176, 255));       //110
SkillMovie.AddBitmap(SASUKESKILL8, RGB(73, 176, 255));       //1
SkillMovie.AddBitmap(SASUKESKILL8, RGB(73, 176, 255));       //2
SkillMovie.AddBitmap(SASUKESKILL8, RGB(73, 176, 255));       //3
SkillMovie.AddBitmap(SASUKERENDAN1, RGB(73, 176, 255));          //4
SkillMovie.AddBitmap(SASUKERENDAN1, RGB(73, 176, 255));          //5
SkillMovie.AddBitmap(SASUKERENDAN1, RGB(73, 176, 255));          //6
SkillMovie.AddBitmap(SASUKERENDAN2, RGB(73, 176, 255));          //7
SkillMovie.AddBitmap(SASUKERENDAN2, RGB(73, 176, 255));          //8
SkillMovie.AddBitmap(SASUKERENDAN2, RGB(73, 176, 255));          //9
SkillMovie.AddBitmap(SASUKERENDAN1, RGB(73, 176, 255));          //120
SkillMovie.AddBitmap(SASUKERENDAN1, RGB(73, 176, 255));          //1
SkillMovie.AddBitmap(SASUKERENDAN1, RGB(73, 176, 255));          //2
SkillMovie.AddBitmap(SASUKERENDAN2, RGB(73, 176, 255));          //3
SkillMovie.AddBitmap(SASUKERENDAN2, RGB(73, 176, 255));          //4
SkillMovie.AddBitmap(SASUKERENDAN2, RGB(73, 176, 255));          //5
SkillMovie.AddBitmap(SASUKERENDAN1, RGB(73, 176, 255));          //6
SkillMovie.AddBitmap(SASUKERENDAN1, RGB(73, 176, 255));          //7
SkillMovie.AddBitmap(SASUKERENDAN1, RGB(73, 176, 255));          //8
```

```
SkillMovie.AddBitmap(SASUKERENDAN2, RGB(73, 176, 255));          //9
SkillMovie.AddBitmap(SASUKERENDAN2, RGB(73, 176, 255));          //130
SkillMovie.AddBitmap(SASUKERENDAN2, RGB(73, 176, 255));          //1
SkillMovie.AddBitmap(SASUKERENDAN1, RGB(73, 176, 255));          //2
SkillMovie.AddBitmap(SASUKERENDAN1, RGB(73, 176, 255));          //3
SkillMovie.AddBitmap(SASUKERENDAN1, RGB(73, 176, 255));          //4
SkillMovie.AddBitmap(SASUKERENDAN2, RGB(73, 176, 255));          //5
SkillMovie.AddBitmap(SASUKERENDAN2, RGB(73, 176, 255));          //6
SkillMovie.AddBitmap(SASUKERENDAN2, RGB(73, 176, 255));          //7
SkillMovie.AddBitmap(SASUKERENDAN22, RGB(73, 176, 255));    //8
SkillMovie.AddBitmap(SASUKERENDAN22, RGB(73, 176, 255));    //9
SkillMovie.AddBitmap(SASUKERENDAN22, RGB(73, 176, 255));    //140
SkillMovie.AddBitmap(SASUKERENDAN23, RGB(73, 176, 255));    //1
SkillMovie.AddBitmap(SASUKERENDAN23, RGB(73, 176, 255));    //2
SkillMovie.AddBitmap(SASUKERENDAN23, RGB(73, 176, 255));    //3
SkillMovie.AddBitmap(SASUKERENDAN22, RGB(73, 176, 255));    //4
SkillMovie.AddBitmap(SASUKERENDAN22, RGB(73, 176, 255));    //5
SkillMovie.AddBitmap(SASUKERENDAN22, RGB(73, 176, 255));    //6
SkillMovie.AddBitmap(SASUKERENDAN23, RGB(73, 176, 255));    //7
SkillMovie.AddBitmap(SASUKERENDAN23, RGB(73, 176, 255));    //8
SkillMovie.AddBitmap(SASUKERENDAN23, RGB(73, 176, 255));    //9
SkillMovie.AddBitmap(SASUKERENDAN22, RGB(73, 176, 255));    //150
SkillMovie.AddBitmap(SASUKERENDAN22, RGB(73, 176, 255));    //1
SkillMovie.AddBitmap(SASUKERENDAN22, RGB(73, 176, 255));    //2
SkillMovie.AddBitmap(SASUKERENDAN23, RGB(73, 176, 255));    //3
SkillMovie.AddBitmap(SASUKERENDAN23, RGB(73, 176, 255));    //4
SkillMovie.AddBitmap(SASUKERENDAN23, RGB(73, 176, 255));    //5
SkillMovie.AddBitmap(SASUKERENDAN22, RGB(73, 176, 255));    //6
SkillMovie.AddBitmap(SASUKERENDAN22, RGB(73, 176, 255));    //7
SkillMovie.AddBitmap(SASUKERENDAN22, RGB(73, 176, 255));    //8
SkillMovie.AddBitmap(SASUKERENDAN23, RGB(73, 176, 255));    //9
SkillMovie.AddBitmap(SASUKERENDAN23, RGB(73, 176, 255));    //160
SkillMovie.AddBitmap(SASUKERENDAN23, RGB(73, 176, 255));    //1
SkillsuccessMovie.AddBitmap(SASUKERENDAN3, RGB(73, 176, 255));          //0
SkillsuccessMovie.AddBitmap(SASUKERENDAN4, RGB(73, 176, 255));          //1
SkillsuccessMovie.AddBitmap(SASUKERENDAN5, RGB(73, 176, 255));          //2
SkillsuccessMovie.AddBitmap(SASUKERENDAN5, RGB(73, 176, 255));          //3
SkillsuccessMovie.AddBitmap(SASUKERENDAN5, RGB(73, 176, 255));          //4
SkillsuccessMovie.AddBitmap(SASUKERENDAN6, RGB(73, 176, 255));          //5
SkillsuccessMovie.AddBitmap(SASUKERENDAN6, RGB(73, 176, 255));          //6
SkillsuccessMovie.AddBitmap(SASUKERENDAN6, RGB(73, 176, 255));          //7
SkillsuccessMovie.AddBitmap(SASUKERENDAN7, RGB(73, 176, 255));          //8
SkillsuccessMovie.AddBitmap(SASUKERENDAN7, RGB(73, 176, 255));          //9
SkillsuccessMovie.AddBitmap(SASUKERENDAN7, RGB(73, 176, 255));          //10
SkillsuccessMovie.AddBitmap(SASUKERENDAN8, RGB(73, 176, 255));          //1
SkillsuccessMovie.AddBitmap(SASUKERENDAN8, RGB(73, 176, 255));          //2
SkillsuccessMovie.AddBitmap(SASUKERENDAN8, RGB(73, 176, 255));          //3
SkillsuccessMovie.AddBitmap(SASUKERENDAN9, RGB(73, 176, 255));          //4
SkillsuccessMovie.AddBitmap(SASUKERENDAN9, RGB(73, 176, 255));          //5
SkillsuccessMovie.AddBitmap(SASUKERENDAN9, RGB(73, 176, 255));          //6
SkillsuccessMovie.AddBitmap(SASUKERENDAN10, RGB(73, 176, 255));         //7
SkillsuccessMovie.AddBitmap(SASUKERENDAN10, RGB(73, 176, 255));         //8
SkillsuccessMovie.AddBitmap(SASUKERENDAN10, RGB(73, 176, 255));         //9
SkillsuccessMovie.AddBitmap(SASUKERENDAN11, RGB(73, 176, 255));         //20
SkillsuccessMovie.AddBitmap(SASUKERENDAN11, RGB(73, 176, 255));         //1
SkillsuccessMovie.AddBitmap(SASUKERENDAN11, RGB(73, 176, 255));         //2
SkillsuccessMovie.AddBitmap(SASUKERENDAN12, RGB(73, 176, 255));         //3
SkillsuccessMovie.AddBitmap(SASUKERENDAN12, RGB(73, 176, 255));         //4
SkillsuccessMovie.AddBitmap(SASUKERENDAN12, RGB(73, 176, 255));         //5
SkillsuccessMovie.AddBitmap(SASUKERENDAN13, RGB(73, 176, 255));         //6
SkillsuccessMovie.AddBitmap(SASUKERENDAN13, RGB(73, 176, 255));         //7
SkillsuccessMovie.AddBitmap(SASUKERENDAN13, RGB(73, 176, 255));         //8
SkillsuccessMovie.AddBitmap(SASUKERENDAN14, RGB(73, 176, 255));         //9
SkillsuccessMovie.AddBitmap(SASUKERENDAN14, RGB(73, 176, 255));         //30
SkillsuccessMovie.AddBitmap(SASUKERENDAN14, RGB(73, 176, 255));         //1
SkillsuccessMovie.AddBitmap(SASUKERENDAN15, RGB(73, 176, 255));         //2
SkillsuccessMovie.AddBitmap(SASUKERENDAN15, RGB(73, 176, 255));         //3
SkillsuccessMovie.AddBitmap(SASUKERENDAN15, RGB(73, 176, 255));         //4
SkillsuccessMovie.AddBitmap(SASUKERENDAN16, RGB(73, 176, 255));         //5
SkillsuccessMovie.AddBitmap(SASUKERENDAN16, RGB(73, 176, 255));         //6
SkillsuccessMovie.AddBitmap(SASUKERENDAN16, RGB(73, 176, 255));         //7
SkillsuccessMovie.AddBitmap(SASUKERENDAN17, RGB(73, 176, 255));         //8
SkillsuccessMovie.AddBitmap(SASUKERENDAN17, RGB(73, 176, 255));         //9
SkillsuccessMovie.AddBitmap(SASUKERENDAN17, RGB(73, 176, 255));         //40
SkillsuccessMovie.AddBitmap(SASUKERENDAN18, RGB(73, 176, 255));         //1
SkillsuccessMovie.AddBitmap(SASUKERENDAN18, RGB(73, 176, 255));         //2
```

```
		SkillsuccessMovie.AddBitmap(SASUKERENDAN18, RGB(73, 176, 255));	//3
		SkillsuccessMovie.AddBitmap(SASUKERENDAN19, RGB(73, 176, 255));	//4
		SkillsuccessMovie.AddBitmap(SASUKERENDAN19, RGB(73, 176, 255));	//5
		SkillsuccessMovie.AddBitmap(SASUKERENDAN19, RGB(73, 176, 255));	//6
		SkillsuccessMovie.AddBitmap(SASUKERENDAN20, RGB(73, 176, 255));	//7
		SkillsuccessMovie.AddBitmap(SASUKERENDAN20, RGB(73, 176, 255));	//8
		SkillsuccessMovie.AddBitmap(SASUKERENDAN20, RGB(73, 176, 255));	//9
		SkillsuccessMovie.AddBitmap(SASUKERENDAN21, RGB(73, 176, 255));	//50
		SkillsuccessMovie.AddBitmap(SASUKERENDAN21, RGB(73, 176, 255));	//1
		SkillsuccessMovie.AddBitmap(SASUKERENDAN21, RGB(73, 176, 255));	//2
		SkillsuccessMovie.AddBitmap(SASUKERENDAN24, RGB(73, 176, 255));	//3
		SkillsuccessMovie.AddBitmap(SASUKERENDAN25, RGB(73, 176, 255));	//4
		SkillsuccessMovie.AddBitmap(SASUKERENDAN26, RGB(73, 176, 255));	//5
		SkillsuccessMovie.AddBitmap(SASUKERENDAN26, RGB(73, 176, 255));	//6
		SkillsuccessMovie.AddBitmap(SASUKERENDAN26, RGB(73, 176, 255));	//7
		SkillsuccessMovie.AddBitmap(SASUKERENDAN27, RGB(73, 176, 255));	//8
		SkillsuccessMovie.AddBitmap(SASUKERENDAN27, RGB(73, 176, 255));	//9
		SkillsuccessMovie.AddBitmap(SASUKERENDAN27, RGB(73, 176, 255));	//60
		SkillsuccessMovie.AddBitmap(SASUKERENDAN28, RGB(73, 176, 255));	//1
		SkillsuccessMovie.AddBitmap(SASUKERENDAN28, RGB(73, 176, 255));	//2
		SkillsuccessMovie.AddBitmap(SASUKERENDAN28, RGB(73, 176, 255));	//3
		SkillsuccessMovie.AddBitmap(SASUKERENDAN29, RGB(73, 176, 255));	//4
		SkillsuccessMovie.AddBitmap(SASUKERENDAN29, RGB(73, 176, 255));	//5
		SkillsuccessMovie.AddBitmap(SASUKERENDAN29, RGB(73, 176, 255));	//6
		SkillsuccessMovie.AddBitmap(SASUKERENDAN30, RGB(73, 176, 255));	//7
		SkillsuccessMovie.AddBitmap(SASUKERENDAN30, RGB(73, 176, 255));	//8
		SkillsuccessMovie.AddBitmap(SASUKERENDAN30, RGB(73, 176, 255));	//9
		SkillsuccessMovie.AddBitmap(SASUKERENDAN31, RGB(73, 176, 255));	//70
		SkillsuccessMovie.AddBitmap(SASUKERENDAN31, RGB(73, 176, 255));	//1
		SkillsuccessMovie.AddBitmap(SASUKERENDAN31, RGB(73, 176, 255));	//2
		SkillsuccessMovie.AddBitmap(SASUKERENDAN32, RGB(73, 176, 255));	//3
		SkillsuccessMovie.AddBitmap(SASUKERENDAN32, RGB(73, 176, 255));	//4
		SkillsuccessMovie.AddBitmap(SASUKERENDAN32, RGB(73, 176, 255));	//5
		SkillsuccessMovie.AddBitmap(SASUKERENDAN33, RGB(73, 176, 255));	//6
		SkillsuccessMovie.AddBitmap(SASUKERENDAN33, RGB(73, 176, 255));	//7
		SkillsuccessMovie.AddBitmap(SASUKERENDAN33, RGB(73, 176, 255));	//8
		SkillsuccessMovie.AddBitmap(SASUKERENDAN34, RGB(73, 176, 255));	//9
		SkillsuccessMovie.AddBitmap(SASUKERENDAN34, RGB(73, 176, 255));	//80
		SkillsuccessMovie.AddBitmap(SASUKERENDAN34, RGB(73, 176, 255));	//1
		SkillsuccessMovie.AddBitmap(SASUKERENDAN35, RGB(73, 176, 255));	//2
		SkillsuccessMovie.AddBitmap(SASUKERENDAN35, RGB(73, 176, 255));	//3
		SkillsuccessMovie.AddBitmap(SASUKERENDAN35, RGB(73, 176, 255));	//4
		SkillsuccessMovie.AddBitmap(SASUKERENDAN36, RGB(73, 176, 255));	//5
		SkillsuccessMovie.AddBitmap(SASUKERENDAN36, RGB(73, 176, 255));	//6
		SkillsuccessMovie.AddBitmap(SASUKERENDAN36, RGB(73, 176, 255));	//7
		SkillsuccessMovie.AddBitmap(SASUKERENDAN37, RGB(73, 176, 255));	//8
		SkillsuccessMovie.AddBitmap(SASUKERENDAN37, RGB(73, 176, 255));	//9
		SkillsuccessMovie.AddBitmap(SASUKERENDAN37, RGB(73, 176, 255));	//90
		SkillsuccessMovie.AddBitmap(SASUKERENDAN38, RGB(73, 176, 255));	//1
		SkillsuccessMovie.AddBitmap(SASUKERENDAN38, RGB(73, 176, 255));	//2
		SkillsuccessMovie.AddBitmap(SASUKERENDAN38, RGB(73, 176, 255));	//3
		SkillsuccessMovie.AddBitmap(SASUKERENDAN39, RGB(73, 176, 255));	//4
		SkillsuccessMovie.AddBitmap(SASUKERENDAN39, RGB(73, 176, 255));	//5
		SkillsuccessMovie.AddBitmap(SASUKERENDAN39, RGB(73, 176, 255));	//6
		SkillsuccessMovie.AddBitmap(SASUKERENDAN40, RGB(73, 176, 255));	//7
		SkillsuccessMovie.AddBitmap(SASUKERENDAN40, RGB(73, 176, 255));	//8
		SkillsuccessMovie.AddBitmap(SASUKERENDAN40, RGB(73, 176, 255));	//9
		SkillsuccessMovie.AddBitmap(SASUKERENDAN41, RGB(73, 176, 255));	//100
		SkillsuccessMovie.AddBitmap(SASUKERENDAN41, RGB(73, 176, 255));	//1
		SkillsuccessMovie.AddBitmap(SASUKERENDAN41, RGB(73, 176, 255));	//2
		SkillsuccessMovie.AddBitmap(SASUKERENDAN42, RGB(73, 176, 255));	//3
		SkillsuccessMovie.AddBitmap(SASUKERENDAN42, RGB(73, 176, 255));	//4
		SkillsuccessMovie.AddBitmap(SASUKERENDAN42, RGB(73, 176, 255));	//5
		Movie.LoadBitmap(SasukeSkillUI, RGB(73, 176, 255));
		Movie2.LoadBitmap(SasukeSkill2UI, RGB(73, 176, 255));
		SkillMovie.SetDelayCount(1);
		SkillMovie.Reset();
		SkillsuccessMovie.SetDelayCount(2);
		SkillsuccessMovie.Reset();
		//////////////////////////////////////////////////////////////////////////////////////////////////
}
void Sasuke::OnMove()
{
		animation.SetDelayCount(1);
		//////////////被攻擊////////////////////////
		if (isAttacked && !isDie && !(animation.GetCurrentBitmapNumber() >= 123 + direction * 140 && animation.GetCurrentBitmapNumber() <=
```

```
        130 + direction * 140) && !isSkill) Attacked();
        else if (Move && !isDie)
        {
                ////////////////////////////////////////使用技能、攻擊、集氣、移動、跳躍、恢復查克拉////////////////////////////////////////////////////////
                if (animation.GetCurrentBitmapNumber() >= 29 + direction * 140 && animation.GetCurrentBitmapNumber() <= 31 + direction * 140)
                { jump(); attacknum = 0;       }
                else if (isSkillFail) SetSkill();
                else if (attacknum > 0) Attack();
                else if (isMolding && ((animation.GetCurrentBitmapNumber() >= 0 + direction * 140 && animation.GetCurrentBitmapNumber() < 11
                + direction * 140) || (animation.GetCurrentBitmapNumber() >= 114 + direction * 140 && animation.GetCurrentBitmapNumber() <=
                130 + direction * 140)))Molding();
                else if (isMovingLeft || isMovingRight || isMovingUp) MovePlayer();
                else Stand();
                if(!isSkill)jump();
                if (chakura <= 64 && chakuradelay-- <= 0 && !isSkillFail) { chakura++; chakuradelay = 16; }
                ////////////////////////////////////////////////////////////////////////////////////////////
        }
        ////////////////////////////////倒地//////////////////////////////////////
        if (isDie) Die();
        ////////////////////////////////投擲物移動//////////////////////////////////
        ThrowMove();
}
void Sasuke::setfloor(int _floor)
{
        floor = _floor; ////////////////////////////設定地板
}
void Sasuke::jump()
{
        ////////////////////////////////////////////跳躍/////////////////////////////////////////
                if (rising)                                     // 上升狀態
                {
                        if (velocity > 0)
                        {
                                y -= velocity;  // 當速度 > 0 時，y 軸上升(移動 velocity 個點，velocity 的單位為 點/次)
                                velocity -= 2;
                        }
                        else
                        {
                                rising = false; // 當速度 <= 0，上升終止，下次改為下降
                                velocity = 1;   // 下降的初速(velocity)為 1
                        }
                }
                else                                    // 下降狀態
                {
                        if (GetY2() < floor)    // 當 y 座標還沒碰到地板
                        {
                                y += velocity; // y 軸下降(移動 velocity 個點，velocity 的單位為 點/次)
                                velocity += 2;
                                if (attacknum == 0) animation.Movetonum(18 + direction * 140);
                        }
                        else
                        {
                                y = floor - 72;   // 當 y 座標低於地板，更正為地板上
                                velocity = initial_velocity; // 重設上升初始速度
                                if (isMovingUp && attacknum == 0)
                                {
                                        ////////////////////////跳躍結束動作////////////////////////////
                                        if (!(animation.GetCurrentBitmapNumber() >= 29 + direction * 140 && animation.GetCurrentBitmapNumber()
                                        <= 31 + direction * 140))animation.Movetonum(29 + direction * 140);
                                        else animation.OnMove();
                                        if (animation.GetCurrentBitmapNumber() == 32 + direction * 140)
                                        {
                                                isMovingUp = false;
                                        }
                                        ////////////////////////////////////////////////////
                                }
                        }
                }
}
void Sasuke::OnShow()
{
        ////////////////////////////顯示//////////////////////////////////
        if (!isSkill)
        {
                animation.SetTopLeft(GetX1(), GetY1());
                animation.OnShow(2);
```

```
		}
		else if(!Skillsuccess)
		{
			SkillMovie.SetTopLeft(GetX1(), GetY1());
			if(SkillMovie.GetCurrentBitmapNumber()>=15+direction * 45 && SkillMovie.GetCurrentBitmapNumber() <= 44+direction * 45)
			SkillMovie.SetTopLeft(GetX1(), GetY1()+14);
			SkillMovie.OnShow(2);
		}
		if (Skillsuccess) SkillsuccessMovie.OnShow(2);
		for (int count = 0; count < (int)Fireball.size(); count++) Fireball[count]->OnShow();
		for (int count = 0; count < (int)shurikan.size(); count++) shurikan[count]->OnShow();
		//////////////////////////////////////////////////////////////////////////
}
void Sasuke::SetMovingLeft(bool flag)
{
		isMovingLeft = flag;
}
void Sasuke::SetMovingRight(bool flag)
{
		isMovingRight = flag;
}
void Sasuke::SetMovingUp(bool flag)
{
		isMovingUp = flag;
}
void Sasuke::SetAttacked(int flag)
{
		isAttacked = flag;
}
void Sasuke::SetDie()
{
		isDie = true;
		CAudio::Instance()->Play(29);
		velocity = 18;
		damage = 1;
}
void Sasuke::SetDie(int Sub)
{
		isDie = true;
		CAudio::Instance()->Play(29);
		velocity = 18;
		damage = Sub;
}
void Sasuke::SetMove(bool flag)
{
		Move = flag;
}
int Sasuke::GetChakura()
{
		return chakura;
}
void Sasuke::ChangeChakura(int _chakura)
{
		chakura = _chakura;
		if (chakura <= 0) chakura = 0;
}
void Sasuke::SetDirection(int _direction)
{
		direction = _direction;
}
void Sasuke::SetFireballDelete(int flag)
{
		delete Fireball[flag];
		Fireball.erase(Fireball.begin() + flag);
}
void Sasuke::SetShurikanDelete(int flag)
{
		delete shurikan[flag];
		shurikan.erase(shurikan.begin() + flag);
}
bool Sasuke::GetSkillsuccess()
{
		return Skillsuccess;
}
void Sasuke::SetSkillsuccess()
{
```

```
		Skillsuccess = true;
	}
	CAnimation * Sasuke::GetSkillsuccessMovie()
	{
		return &SkillsuccessMovie;
	}
	void Sasuke::RunSkillsuccessMovie()
	{
		/////////////////////////////////////////////////////技能成功動畫/////////////////////////////////////////////////////
		if (SkillsuccessMovie.GetCurrentBitmapNumber() == 52 + direction * 53)
		{
			isSkill = false;
			Skillsuccess = false;
			SkillsuccessMovie.Reset();
			Movieflag = false;
			isSkillFail = false;
			isAttacked = false;
			magic = 0;
		}
		else SkillsuccessMovie.OnMove();
		if(SkillsuccessMovie.GetCurrentBitmapNumber() == 17 + direction * 53) CAudio::Instance()->Play(27);
		SkillsuccessMovie.SetTopLeft(GetX1(), GetY1());
		if(SkillsuccessMovie.GetCurrentBitmapNumber() >= 14 + direction * 53 && SkillsuccessMovie.GetCurrentBitmapNumber() < 32 + direction *
53) y -= 5;
		if(SkillsuccessMovie.GetCurrentBitmapNumber() > 32 + direction * 53) y += 5;
		////////////////////////////////////////////////////////////////////////////////////////////////////////////////
	}
	void Sasuke::Deleteobject()
	{
		/////////////////////////////////////////////刪除子彈/////////////////////////////////////////////////////////////
		int num = (int)Fireball.size();
		for (int i = 0; i < num; i++)
		{
			delete Fireball[0];
			Fireball.erase(Fireball.begin());
		}
		num = (int)shurikan.size();
		for (int i = 0; i < num; i++)
		{
			delete shurikan[0];
			shurikan.erase(shurikan.begin());
		}
		////////////////////////////////////////////////////////////////////////////////////////////////////////////////
	}
	void Sasuke::Attack()
	{
		/////////////////////////////////////////////////////攻擊動畫/////////////////////////////////////////////////////
		if (GetY2() >= floor && !(animation.GetCurrentBitmapNumber() >= 20 + direction * 140 && animation.GetCurrentBitmapNumber() < 28 +
		direction * 140))
		{
			if (!(animation.GetCurrentBitmapNumber() >= 32 + direction * 140 && animation.GetCurrentBitmapNumber() < 61 + direction * 140))
			{
				if (rand() % 2 == 0) CAudio::Instance()->Play(21);
				else CAudio::Instance()->Play(22);
				CAudio::Instance()->Play(23);
				animation.Movetonum(32 + direction * 140);
			}
			else
				animation.OnMove();
			if (animation.GetCurrentBitmapNumber() == 40 + direction * 140 || animation.GetCurrentBitmapNumber() == 49 + direction * 140)
			{
				if (direction == 1 && GetX1() - 25 >= 0) x -= 12;
				else if (direction == 0 && GetX2() + 25 <= 640)x += 12;
				attacknum--;
			}
			else if (animation.GetCurrentBitmapNumber() == 61 + direction * 140)
			{
				attacknum = 0;
				if (direction == 1 && GetX1() - 25 >= 0) x -= 25;
				else if (direction == 0 && GetX2() + 25 <= 640)x += 25;
			}
			if (animation.GetCurrentBitmapNumber() == 41 + direction * 140 || animation.GetCurrentBitmapNumber() == 50 + direction * 140)
			CAudio::Instance()->Play(23);
		}
		/////////////////////////////////////////////////////////跳躍攻擊/////////////////////////////////////////////////////
		else
```

```
	{
		if (!(animation.GetCurrentBitmapNumber() >= 20 + direction * 140 && animation.GetCurrentBitmapNumber() < 28 + direction * 140))
		{
			if (rand() % 2 == 0) CAudio::Instance()->Play(21);
			else CAudio::Instance()->Play(22);
			CAudio::Instance()->Play(23);
			animation.Movetonum(20 + direction * 140);
		}
		else
			animation.OnMove();
		if (animation.GetCurrentBitmapNumber() == 28 + direction * 140)
			attacknum = 0;
		if (isMovingLeft && GetX1() - step_size >= 0) x -= step_size;
		if (isMovingRight&& GetX2() + step_size <= 640) x += step_size;
	}
	//////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////
}
void Sasuke::Attacked()
{
	//////////////////////////////////////////////////////////////被攻擊//////////////////////////////////////////////////////////
	if (!(animation.GetCurrentBitmapNumber() >= 111 + direction * 140 && animation.GetCurrentBitmapNumber() < 113 + direction * 140))
	{
		animation.Movetonum(111 + direction * 140);
		CAudio::Instance()->Play(29);
	}
	else
		animation.OnMove();
	//////////////////////////////////////////////////////////////角色擊退//////////////////////////////////////////////////////////
	if (direction == 1 && GetX2() + 4 <= 640) x += 4;
	else if (direction == 0 && GetX1() - 4 >= 0)x -= 4;
	//////////////////////////////////////////////////////////////減少血量//////////////////////////////////////////////////////////
	if (animation.GetCurrentBitmapNumber() == 113 + direction * 140) isAttacked = false;
	if (Blood > 0)Blood -= 2;		//減少血量
	attacknum = 0;
	//////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////
}
void Sasuke::Molding()
{
	//////////////////////////////////////////////////////////////集氣動畫//////////////////////////////////////////////////////////
	if (Moldingdelay-- <= 0)
	{
		if (!(animation.GetCurrentBitmapNumber() >= 114 + direction * 140 && animation.GetCurrentBitmapNumber() <= 130 + direction *
		140))
		{
			animation.Movetonum(114 + direction * 140);
			CAudio::Instance()->Play(12, true);
		}
		else if (animation.GetCurrentBitmapNumber() == 130 + direction * 140)
		{
			animation.Movetonum(123 + direction * 140);
			isAttacked = false;
		}
		else
			animation.OnMove();
		if ((animation.GetCurrentBitmapNumber() >= 123 + direction * 140 && animation.GetCurrentBitmapNumber() <= 130 + direction *
		140))
		{
			magic += 8;
			chakura -= 1;
			isAttacked = false;
			//////////////////////////////////////////////////集氣失敗//////////////////////////////////////////////////////////
			if (magic >= 512 || chakura < 0)
			{
				isSkillFail = true;
				isMolding = false;
			}
		}
	}
	//////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////
}
void Sasuke::ThrowMove()
{
	for (int count = 0; count < (int)Fireball.size(); count++) Fireball[count]->Move();
	for (int count = 0; count < (int)shurikan.size(); count++) shurikan[count]->Move();
	//////////////////////////////////////////////////////////刪除超過邊界之投擲物//////////////////////
	for (int count = 0; count < (int)Fireball.size(); count++)
```

```
	{
		Fireball[count]->Move();
		if ((Fireball[count]->GetX1() > 640 && Fireball[count]->Getdirection() == 0) || ((Fireball[count]->GetX1() > 640 || Fireball[count]->GetY1() > 480) && Fireball[count]->Getdirection() == 1) || (Fireball[count]->GetY1() > 480 && Fireball[count]->Getdirection() == 2) || ((Fireball[count]->GetX2() < 0 || Fireball[count]->GetY1() > 480) && Fireball[count]->Getdirection() == 3) || (Fireball[count]->GetX2() < 0 && Fireball[count]->Getdirection() == 4) || ((Fireball[count]->GetX2() < 0 || Fireball[count]->GetY2() < 0) && Fireball[count]->Getdirection() == 5) || (Fireball[count]->GetY2() < 0 && Fireball[count]->Getdirection() == 6) || ((Fireball[count]->GetX1() > 640 || Fireball[count]->GetY2() < 0) && Fireball[count]->Getdirection() == 7))
		{
			delete Fireball[count];
			Fireball.erase(Fireball.begin() + count);
		}
	}
	for (int count = 0; count < (int)shurikan.size(); count++)
	{
		shurikan[count]->Move();
		if ((shurikan[count]->GetX2() < 0 && direction == 0) || (shurikan[count]->GetX1() > 640 && direction == 1))
		{
			delete shurikan[count];
			shurikan.erase(shurikan.begin() + count);
		}
	}
	////////////////////////////////////////////////////////////////////////////////////////////////////
}
void Sasuke::MovePlayer()
{
	//////////////////////////////////////////////////左移//////////////////////////////////////////////
	if (isMovingLeft)
	{
		if (GetX1() > 0)
			x -= step_size;
		if (GetY2() >= floor)
		{
			if (!(animation.GetCurrentBitmapNumber() >= 12 + direction * 140 && animation.GetCurrentBitmapNumber() < 17 + direction * 140))
			{
				animation.Movetonum(12 + direction * 140);
			}
			else
			{
				animation.OnMove();
			}
		}
		direction = 1;
	}
	//////////////////////////////////////////////////右移//////////////////////////////////////////////
	else if (isMovingRight)
	{
		if (GetX2() < 640)
			x += step_size;
		if (GetY2() >= floor)
		{
			if (!(animation.GetCurrentBitmapNumber() >= 12 && animation.GetCurrentBitmapNumber() < 17))
			{
				animation.Movetonum(12);
			}
			else
			{
				animation.OnMove();
			}
		}
		direction = 0;
	}
	//////////////////////////////////////////////////跳躍//////////////////////////////////////////////
	if ((isMovingUp && rising == false && velocity == initial_velocity && GetY2() >= floor) && !(animation.GetCurrentBitmapNumber() >= 29 + direction * 140 && animation.GetCurrentBitmapNumber() <= 31 + direction * 140))
	{
		CAudio::Instance()->Play(23);
		rising = true;
		velocity = initial_velocity;
		if (isMovingRight || isMovingLeft)
			animation.Movetonum(19 + direction * 140);
		else
			animation.Movetonum(18 + direction * 140);

	}
```

```
        ////////////////////////////////////////////////////////////////////////////////////////////
    }
    void Sasuke::Die()
    {
        ////////////////////////////////////人物倒地////////////////////////////////////////////////////
        Move = false;
        if (animation.GetCurrentBitmapNumber() < 62 + direction * 140 || animation.GetCurrentBitmapNumber() > 110 + direction * 140)
        animation.Movetonum(62 + direction * 140);
        if (animation.GetCurrentBitmapNumber() >= 74 + direction * 140 && animation.GetCurrentBitmapNumber() <= 79 + direction * 140)
        {
            velocity -= 3;
            if (GetY2() < floor) y -= velocity;     // 當後半段動畫角色從空中落下
        }
        if (animation.GetCurrentBitmapNumber() >= 62 + direction * 140 && animation.GetCurrentBitmapNumber() <= 67 + direction * 140)
        {
            y -= velocity;  // 當前半段動畫角色被擊飛至空中
            velocity -= 3;
        }
        if (animation.GetCurrentBitmapNumber() >= 62 + direction * 140 && animation.GetCurrentBitmapNumber() <= 79 + direction * 140)
        {
            ////////////////////////////////////角色往後飛//////////////////////////////////
            if (direction == 1 && GetX2() + 7 <= 640) x += 7;
            else if (direction == 0 && GetX1() - 7 >= 0) x -= 7;
            ////////////////////////////////////////////////////////////////////////////////
        }
        if (GetY2() < floor) y -= velocity;//////////////////確保不超過地面且到達地面
        if (animation.GetCurrentBitmapNumber() == 80 + direction * 140) CAudio::Instance()->Play(30);
        if (animation.GetCurrentBitmapNumber() == 110 + direction * 140)
        {
            ////////////////////////////////倒地延遲//////////////////////////////////
            if (Diedelay-- <= 0)
            {
                isDie = false;
                Move = true;
                isAttacked = false;
                Diedelay = 30;
                velocity = initial_velocity;
                y = floor - 72;
            }
            ////////////////////////////////////////////////////////////////////////
        }
        else    animation.OnMove();
        ////////////////////////////////減少血量////////////////////////////////////
        if (Blood > 0 && delay-- <= 0 && animation.GetCurrentBitmapNumber() != 110 + direction * 140)
        {
            Blood -= damage;
            delay = 2;
        }
        ////////////////////////////////////////////////////////////////////////////
    }
    void Sasuke::Stand()
    {
        ////////////////////////////////站在原地//////////////////////////////////////
        if (!(animation.GetCurrentBitmapNumber() >= 0 + direction * 140 && animation.GetCurrentBitmapNumber() < 11 + direction * 140))
            animation.Movetonum(0 + direction * 140);
        else
            animation.OnMove();
        if (chakura <= 64 && chakuradelay-- <= 8) { chakura++; chakuradelay = 16; }
        ////////////////////////////////////////////////////////////////////////////////
    }
    void Sasuke::SetSkill()
    {
        ////////////////////////////////////////技能判定失敗/////////////////////////////////////////
        CAudio::Instance()->Stop(12);
        if (magic >= 512 || chakura <= 0)
        {
            if (!(animation.GetCurrentBitmapNumber() >= 131 + direction * 140 && animation.GetCurrentBitmapNumber() < 139 + direction *
            140))
                animation.Movetonum(131 + direction * 140);
            else
                animation.OnMove();
            if (animation.GetCurrentBitmapNumber() == 139 + direction * 140)
            {
                magic = 0;
                isSkillFail = false;
            }
```

```
	}
	//////////////////////////////////////////////技能成功//////////////////////////////////////////////
	else
	{
		if (!Movieflag)
		{
			if (magic >= 0 && magic < 128)		SkillMovie.Movetonum(90 + direction * 12);	//技能一成功
			if (magic >= 128 && magic < 256)	SkillMovie.Movetonum(0 + direction * 45); //技能二成功
			if (magic >= 256 && magic < 384)	SkillMovie.Movetonum(90 + direction * 12);	//技能三成功
			if (magic >= 384 && magic < 512)	SkillMovie.Movetonum(114 + direction * 24);	//技能四成功
			animation.Movetonum(0 + direction * 140);
			Movieflag = true;
			//////////////////////////技能成功//////////////////////
			if (magic % 128 < 38) Movie.Init();
			else
			{
				////////////////技能大成功//////////////////////
				Movie2.Init();
				CAudio::Instance()->Play(24);
				//////////////////////////////////////////////////
			}
			SkillsuccessMovie.SetTopLeft(GetX1(), GetY1());
		}
		isMovingRight = false;
		isMovingLeft = false;
		isAttacked = false;
		isDie = false;
		isSkill = true;
		//////////////////////////////技能動畫//////////////////////////////
		if (magic >= 0 && magic < 128)		AttackMovie1();
		if (magic >= 128 && magic < 256)	AttackMovie2();
		if (magic >= 256 && magic < 384)	AttackMovie3();
		if (magic >= 384 && magic < 512)	AttackMovie4();
		//////////////////////////////////////////////////////////////////
	}
}
void Sasuke::SetisMolding(bool flag)
{
	isMolding = flag;
	Moldingdelay = 3;
	if (!flag &&(animation.GetCurrentBitmapNumber() >= 114 + direction * 140 && animation.GetCurrentBitmapNumber() <=130 + direction *
	140))
	{
		isSkillFail = true;
	}
}
bool Sasuke::GetMovingLeft()
{
	return isMovingLeft;
}
bool Sasuke::GetMovingRight()
{
	return isMovingRight;
}
bool Sasuke::GetisDie()
{
	return isDie;
}
void Sasuke::SubBlood(int Sub)
{
	Blood -= Sub;
}
int Sasuke::GetDirection()
{
	return direction;
}
int Sasuke::GetAttack()
{
	return attacknum;
}
bool Sasuke::GetisSkill()
{
	return isSkill;
}
bool Sasuke::GetisMolding()
{
```

```
	return isMolding;
}
void Sasuke::SetXY(int nx, int ny)
{
	x = nx; y = ny;
}
void Sasuke::Addattacknum()
{
	attacknum++;
}
bool Sasuke::GetMove()
{
	return Move;
}
int Sasuke::Getblood()
{
	return Blood;
}
int Sasuke::GetMagic()
{
	return magic;
}
int Sasuke::GetCurrentBitmapNumber()
{
	return animation.GetCurrentBitmapNumber();
}
int Sasuke::GetSkillCurrentBitmapNumber()
{
	return SkillMovie.GetCurrentBitmapNumber();
}
skillstart *Sasuke::Getskillstart()
{
	if (Movie.isStart()) return &Movie;
	else return &Movie2;
}
vector<SasukeFireball*> Sasuke::GetFireball()
{
	return Fireball;
}
vector<Shurikan*> Sasuke::GetShurikan()
{
	return shurikan;
}
void Sasuke::AttackMovie1()
{
	//////////////////////播放技能前置動畫////////////////////////////////
	if (Movie.isStart())	Movie.OnMove();
	else if (Movie2.isStart())	Movie2.OnMove();
	else
	{
		////////////////////////////////////技能內容//////////////////////////////////////////////////
		if (SkillMovie.GetCurrentBitmapNumber() == 101 + direction * 12)
		{
			//////////////////////////////技能小成功往前射出手里劍//////////////////////////////////////////
			if (direction == 0)
			{
				shurikan.push_back(new Shurikan((this->GetX2() + GetX1()) / 2, (this->GetY2() - this->GetY1()) / 3 + this->GetY1(),
				1));
				shurikan.push_back(new Shurikan((this->GetX2() + GetX1()) / 2, (this->GetY2() - this->GetY1()) / 2 + this->GetY1(),
				1));
			}
			if (direction == 1)
			{
				shurikan.push_back(new Shurikan((this->GetX2() + GetX1()) / 2, (this->GetY2() - this->GetY1()) / 3 + this->GetY1(),
				0));
				shurikan.push_back(new Shurikan((this->GetX2() + GetX1()) / 2, (this->GetY2() - this->GetY1()) / 2 + this->GetY1(),
				0));
			}
			//////////////////////////////技能大成功往前射出一堆手里劍//////////////////////////////////////////
			if (magic % 128 > 38 && skill1delay-- <= 0)
			{
				if (direction == 0)
				{
					shurikan.push_back(new Shurikan((this->GetX2() + GetX1()) / 2, (this->GetY2() - this->GetY1()) / 3 + this-
					>GetY1(), 1));
					shurikan.push_back(new Shurikan((this->GetX2() + GetX1()) / 2, (this->GetY2() - this->GetY1()) / 2 + this-
```

```
                                        >GetY1(), 1));
                                }
                                if (direction == 1)
                                {
                                        shurikan.push_back(new Shurikan((this->GetX2() + GetX1()) / 2, (this->GetY2() - this->GetY1()) / 3 + this-
                                        >GetY1(), 0));
                                        shurikan.push_back(new Shurikan((this->GetX2() + GetX1()) / 2, (this->GetY2() - this->GetY1()) / 2 + this-
                                        >GetY1(), 0));
                                }
                        }
                        ////////////////////////////////////////////////////////////////////////////
                        if (magic % 128 <= 38 || (magic % 128 > 38 && skill1delay <= 0))
                        {
                                CAudio::Instance()->Play(26);
                                isSkill = false;
                                Movieflag = false;
                                isSkillFail = false;
                                isAttacked = false;
                                magic = 0;
                                skill1delay = 8;
                        }
                }
                ////////////////////////////////////////////////////////////////////////////
                else    SkillMovie.OnMove();
        }
}
void Sasuke::AttackMovie2()
{
        //////////////////////播放技能前置動畫//////////////////////////////
        if (Movie.isStart())    Movie.OnMove();
        else if (Movie2.isStart())      Movie2.OnMove();
        else
        {
                ////////////////////////////////////////////技能成功//////////////////////////////////////////
                if(SkillMovie.GetCurrentBitmapNumber() == 20 + direction * 45) CAudio::Instance()->Play(26);
                SkillMovie.OnMove();
                ////////////////////////////////////////////技能大成功//////////////////////////////////////////
                if (magic%128 > 38)
                {
                        ////////////////////////////////////////////射出火球//////////////////////////////////////
                                if (SkillMovie.GetCurrentBitmapNumber() == 20 + direction * 45 || SkillMovie.GetCurrentBitmapNumber() == 26 +
                                direction * 45 || SkillMovie.GetCurrentBitmapNumber() == 32 + direction * 45 || SkillMovie.GetCurrentBitmapNumber()
                                == 38 + direction * 45)
                        {
                                if(direction == 0) Fireball.push_back(new SasukeFireball(this->GetX2()-20, (this->GetY2() - this->GetY1()) / 3 + this-
                                >GetY1(), 0));
                                if(direction == 1) Fireball.push_back(new SasukeFireball(this->GetX1()+20, (this->GetY2() - this->GetY1()) / 3 + this-
                                >GetY1(), 4));
                        }
                        ////////////////////////////////////////////////////////////////////////////////////
                }
                if (SkillMovie.GetCurrentBitmapNumber() == 44 + direction * 45)
                {
                        isSkill = false;
                        Movieflag = false;
                        isSkillFail = false;
                        isAttacked = false;
                        magic = 0;
                }
                ////////////////////////////////////////////////////////////////////////////////////////////////
        }
}
void Sasuke::AttackMovie3()
{
        //////////////////////播放技能前置動畫//////////////////////////////
        if (Movie.isStart())    Movie.OnMove();
        else if (Movie2.isStart())      Movie2.OnMove();
        else
        {
                if (SkillMovie.GetCurrentBitmapNumber() == 101 + direction * 12)
                {
                        if (skill3delay-- <= 0)
                        {
                                ////////////////////////////順時針射出火球//////////////////////////////////////
                                if (direction == 0) Fireball.push_back(new SasukeFireball((this->GetX2() + GetX1()) / 2, (this->GetY2() - this->GetY1())
                                / 3 + this->GetY1(), skill3direction % 8));
```

```
                                if (direction == 1) Fireball.push_back(new SasukeFireball((this->GetX2() + GetX1()) / 2, (this->GetY2() - this->GetY1())
                                / 3 + this->GetY1(), (4 + skill3direction) % 8));
                                ////////////////////////////////////////////////////////////////////////////
                                if (skill3direction > 24)
                                {
                                        isSkill = false;
                                        Movieflag = false;
                                        isSkillFail = false;
                                        isAttacked = false;
                                        magic = 0;
                                }
                                if ((skill3direction >= 23 && magic % 128 > 38) || (skill3direction >= 15 && magic % 128 <= 38))
                                {
                                        isSkill = false;
                                        Movieflag = false;
                                        isSkillFail = false;
                                        isAttacked = false;
                                        magic = 0;
                                        skill3direction = 0;
                                }
                                skill3direction++;
                                skill3delay = 4;
                        }
                }
                else    SkillMovie.OnMove();
                y = 200;
                x = 290;
        }
    }
    void Sasuke::AttackMovie4()
    {
        ////////////////////////播放技能前置動畫////////////////////////////////
        if (Movie.isStart())    Movie.OnMove();
        else if (Movie2.isStart())      Movie2.OnMove();
        else
        {
            ////////////////////////////////////技能動畫////////////////////////////
            if (!Skillsuccess)
            {
                SkillMovie.OnMove();
                if (GetX2() < 640 && direction == 0) x += step_size;
                if (GetX1() > 0 && direction == 1) x -= step_size;
            }
            else RunSkillsuccessMovie();///////技能命中////////////////////////////////
            if (SkillMovie.GetCurrentBitmapNumber() == 137 + direction * 24)
            {
                isSkill = false;
                Movieflag = false;
                isSkillFail = false;
                isAttacked = false;
                magic = 0;
            }
        }
    }
}
```

SasukeFireball.h

```
#ifndef SASUKEFIREBALL_H
#define SASUKEFIREBALL_H
#include "Shoot.h"
namespace game_framework {
    class SasukeFireball : public Shoot
    {
    public:
        SasukeFireball(int _x, int _y, int _direction);
        void Init(int _x, int _y, int _direction);
        int GetX1();                                    // 角色左上角 x 座標
        int GetY1();                                    // 角色左上角 y 座標
        int GetX2();
        int GetY2();
        int Getdirection();
        void AddPic();                          // 載入圖形
        void OnShow();
        void Move();
        void SetXY(int _x, int _y);
    };
```

```
}
#endif
```

SasukeFireball.cpp

```
#include "stdafx.h"
#include "Resource.h"
#include <mmsystem.h>
#include <ddraw.h>
#include "audio.h"
#include "gamelib.h"
#include "SasukeFireball.h"
namespace game_framework {
	SasukeFireball::SasukeFireball(int _x, int _y, int _direction)
	{
		Init(_x, _y, _direction);
		CAudio::Instance()->Play(25);
	}
	void SasukeFireball::Init(int _x, int _y, int _direction)
	{
		SetXY(_x, _y);
		direction = _direction;
		AddPic();
	}
	int SasukeFireball::GetX1()
	{
		return x;
	}
	int SasukeFireball::GetY1()
	{
		return y;
	}
	int SasukeFireball::GetX2()
	{
		return x + Fireballpic.Width() * 2;
	}
	int SasukeFireball::GetY2()
	{
		return y+ Fireballpic.Height()*2;
	}
	int SasukeFireball::Getdirection()
	{
		return direction;
	}
	void SasukeFireball::AddPic()
	{
		//////////////////////////////////////////設定火球各方向圖片//////////////////////////////////////////////////////
		if (direction == 0)
		{
			Fireballpic.AddBitmap(SASUKEFIREBALL1, RGB(73, 176, 255));
			Fireballpic.AddBitmap(SASUKEFIREBALL2, RGB(73, 176, 255));
			Fireballpic.AddBitmap(SASUKEFIREBALL3, RGB(73, 176, 255));
			Fireballpic.AddBitmap(SASUKEFIREBALL4, RGB(73, 176, 255));
			Fireballpic.AddBitmap(SASUKEFIREBALL5, RGB(73, 176, 255));
		}
		if (direction == 1)
		{
			Fireballpic.AddBitmap(SASUKEFIREBALL6, RGB(73, 176, 255));
			Fireballpic.AddBitmap(SASUKEFIREBALL7, RGB(73, 176, 255));
			Fireballpic.AddBitmap(SASUKEFIREBALL8, RGB(73, 176, 255));
			Fireballpic.AddBitmap(SASUKEFIREBALL9, RGB(73, 176, 255));
			Fireballpic.AddBitmap(SASUKEFIREBALL10, RGB(73, 176, 255));
		}
		if (direction == 2)
		{
			Fireballpic.AddBitmap(SASUKEFIREBALL11, RGB(73, 176, 255));
			Fireballpic.AddBitmap(SASUKEFIREBALL12, RGB(73, 176, 255));
			Fireballpic.AddBitmap(SASUKEFIREBALL13, RGB(73, 176, 255));
			Fireballpic.AddBitmap(SASUKEFIREBALL14, RGB(73, 176, 255));
			Fireballpic.AddBitmap(SASUKEFIREBALL15, RGB(73, 176, 255));
		}
		if (direction == 3)
		{
			Fireballpic.AddBitmap(SASUKEFIREBALL16, RGB(73, 176, 255));
			Fireballpic.AddBitmap(SASUKEFIREBALL17, RGB(73, 176, 255));
			Fireballpic.AddBitmap(SASUKEFIREBALL18, RGB(73, 176, 255));
			Fireballpic.AddBitmap(SASUKEFIREBALL19, RGB(73, 176, 255));
			Fireballpic.AddBitmap(SASUKEFIREBALL20, RGB(73, 176, 255));
```

```
		}
		if (direction == 4)
		{
			Fireballpic.AddBitmap(SASUKEFIREBALL21, RGB(73, 176, 255));
			Fireballpic.AddBitmap(SASUKEFIREBALL22, RGB(73, 176, 255));
			Fireballpic.AddBitmap(SASUKEFIREBALL23, RGB(73, 176, 255));
			Fireballpic.AddBitmap(SASUKEFIREBALL24, RGB(73, 176, 255));
			Fireballpic.AddBitmap(SASUKEFIREBALL25, RGB(73, 176, 255));
		}
		if (direction == 5)
		{
			Fireballpic.AddBitmap(SASUKEFIREBALL26, RGB(73, 176, 255));
			Fireballpic.AddBitmap(SASUKEFIREBALL27, RGB(73, 176, 255));
			Fireballpic.AddBitmap(SASUKEFIREBALL28, RGB(73, 176, 255));
			Fireballpic.AddBitmap(SASUKEFIREBALL29, RGB(73, 176, 255));
			Fireballpic.AddBitmap(SASUKEFIREBALL30, RGB(73, 176, 255));
		}
		if (direction == 6)
		{
			Fireballpic.AddBitmap(SASUKEFIREBALL31, RGB(73, 176, 255));
			Fireballpic.AddBitmap(SASUKEFIREBALL32, RGB(73, 176, 255));
			Fireballpic.AddBitmap(SASUKEFIREBALL33, RGB(73, 176, 255));
			Fireballpic.AddBitmap(SASUKEFIREBALL34, RGB(73, 176, 255));
			Fireballpic.AddBitmap(SASUKEFIREBALL35, RGB(73, 176, 255));
		}
		if (direction == 7)
		{
			Fireballpic.AddBitmap(SASUKEFIREBALL36, RGB(73, 176, 255));
			Fireballpic.AddBitmap(SASUKEFIREBALL37, RGB(73, 176, 255));
			Fireballpic.AddBitmap(SASUKEFIREBALL38, RGB(73, 176, 255));
			Fireballpic.AddBitmap(SASUKEFIREBALL39, RGB(73, 176, 255));
			Fireballpic.AddBitmap(SASUKEFIREBALL40, RGB(73, 176, 255));
		}
		//////////////////////////////////////////////////////////////////////////////////
		Fireballpic.SetDelayCount(3);
		Fireballpic.Reset();
	}
	void SasukeFireball::OnShow()
	{
		Fireballpic.SetTopLeft(x, y);
		Fireballpic.OnShow(2);
	}
	void SasukeFireball::Move()
	{
		////////////////////////////////////////////往各方向移動/////////////////////////////////
		if (direction == 0)	x += 15;                                        //往右
		else if (direction == 1) { x += 15; y += 15;}  //往右下
		else if (direction == 2) { y += 15; }                    //下
		else if (direction == 3) { x -= 15; y += 15;}  //往左下
		else if (direction==4) x -= 15;                                 //往左
		else if (direction==5) { x -= 15; y -= 15; }    //往左上
		else if (direction==6) {y -= 15; }                              //往上
		else if (direction==7) { x += 15; y -= 15; }   //往右下
		Fireballpic.OnMove();
	}
	void SasukeFireball::SetXY(int _x, int _y)
	{
		x = _x;
		y = _y;
	}
}
```

SkillStart.h

```
#ifndef SKILLSTART_H
#define SKILLSTART_H
namespace game_framework {
	class skillstart {
	public:
		skillstart();
		void Init();
		void LoadBitmap(int IDB_BITMAP, COLORREF colorkey);
		void OnShow();
		void OnMove();
		void isEnd();
		bool isStart();
	private:
		bool Stop, Start;
```

```
		CMovingBitmap image;
		int delaycount, x, y;
	};
}
#endif
```

SkillStart.cpp

```
#include "stdafx.h"
#include "Resource.h"
#include <mmsystem.h>
#include <ddraw.h>
#include "audio.h"
#include "gamelib.h"
#include "SkillStart.h"
namespace game_framework {
	skillstart::skillstart()
	{
		Init();
		Start = false;
	}
	void skillstart::Init()
	{
		x = 640;
		y = 100;
		Start = true;
		Stop = true;
	}
	/////////////////////////////////////////讀取圖片/////////////////////////////////////////
	void skillstart::LoadBitmap(int IDB_BITMAP, COLORREF colorkey)
	{
		image.LoadBitmap(IDB_BITMAP, colorkey);
	}
	/////////////////////////////////////////顯示技能圖片/////////////////////////////////////
	void skillstart::OnShow()
	{
		image.SetTopLeft(x, y);
		image.ShowBitmap();
		isEnd();
	}
	/////////////////////////////////////////移動技能圖片/////////////////////////////////////
	void skillstart::OnMove()
	{
		if (x > 0) x -= 40;
		else if (x == 0 && Stop)
		{
			delaycount = 40;
			Stop = false;
		}
		else if (delaycount-- <= 0) x -= 40;
	}
	/////////////////////////////////////////////////////////////////////////////////////////
	void skillstart::isEnd()
	{
		if (x <= -640)
			Start = false;
	}
	bool skillstart::isStart()
	{
		return Start;
	}
}
```

Shurikan.h

```
#ifndef SHURIKAN_H
#define SHURIKAN_H
#include "Shoot.h"
namespace game_framework {
	class Shurikan : public Shoot
	{
	public:
		Shurikan(int _x, int _y, int _direction);
		void Init(int _x, int _y, int _direction);
		int GetX1();                                    // 角色左上角 x 座標
		int GetY1();                                    // 角色左上角 y 座標
		int GetX2();
		int GetY2();
		int Getdirection();
		void AddPic();                          // 載入圖形
```

```
		void OnShow();
		void Move();
		void SetXY(int _x, int _y);
	};
}
#endif
```

Shurikan.cpp

```
#include "stdafx.h"
#include "Resource.h"
#include <mmsystem.h>
#include <ddraw.h>
#include "audio.h"
#include "gamelib.h"
#include "Shurikan.h"
namespace game_framework {
	Shurikan::Shurikan(int _x, int _y, int _direction)
	{
		Init(_x, _y, _direction);
	}
	void Shurikan::Init(int _x, int _y, int _direction)
	{
		x = _x;
		y = _y;
		direction = _direction;
		AddPic();
	}
	int Shurikan::GetX1()
	{
		return x;
	}
	int Shurikan::GetY1()
	{
		return y;
	}
	int Shurikan::GetX2()
	{
		return x+ Shurikanpic.Width()*2;
	}
	int Shurikan::GetY2()
	{
		return y+ Shurikanpic.Height()*2;
	}
	int Shurikan::Getdirection()
	{
		return direction;
	}
	void Shurikan::AddPic()
	{
		Shurikanpic.AddBitmap(SHURIKAN, RGB(73, 176, 255));
		Shurikanpic.AddBitmap(SHURIKAN2, RGB(73, 176, 255));
		Shurikanpic.AddBitmap(SHURIKAN3, RGB(73, 176, 255));
		Shurikanpic.SetDelayCount(3);
		Shurikanpic.Reset();
	}
	void Shurikan::OnShow()
	{
		Shurikanpic.SetTopLeft(x, y);
		Shurikanpic.OnShow(2);
	}
	void Shurikan::Move()
	{
		if (direction == 1)x += 15;
		else x -= 15;
		Shurikanpic.OnMove();
	}
	void Shurikan::SetXY(int _x, int _y)
	{
		x = _x;
		y = _y;
	}
}
```

bloodcount.h

```
#ifndef BLOODCOUNT_H
#define BLOODCOUNT_H

namespace game_framework {
```

```
	class bloodcount
	{
	public:
		bloodcount();
		void Init(int _x, int _y, string _characteristic, int _numOfBlood, int IDB_BITMAP, COLORREF colorkey);
		int	GetX1();				// 角色左上角 x 座標
		int	GetY1();				// 角色左上角 y 座標
		void OnShow();
		void SetXY(int _x, int _y);
		void ChangeBlood(int _numOfBlood);
		void SetisMolding(bool flag, int _numOfMagic);
		void ChangeChakura(int _chakura);
		int GetChakura();
		int GetBlood();
	protected:
		CMovingBitmap Blood,Blood2,Magic,Chakura;
		CMovingBitmap characterUI;
		CMovingBitmap MagicBar;
		CMovingBitmap bloodbackground;
		CAnimation SkillText;
		bool isMolding;
		int x, y, numOfBlood,numOfMagic, numOfChakura;
		string characteristic;
	};
}
#endif
```

bloodcount.cpp

```
#include "stdafx.h"
#include "Resource.h"
#include <mmsystem.h>
#include <ddraw.h>
#include "audio.h"
#include "gamelib.h"
#include "bloodcount.h"
namespace game_framework {
	bloodcount::bloodcount()
	{

	}
	void bloodcount::Init(int _x, int _y, string _characteristic, int _numOfBlood, int IDB_BITMAP, COLORREF colorkey)
	{
		/////////////////////////////////////初始化/////////////////////////////////////
		characterUI.LoadBitmap(IDB_BITMAP, colorkey);
		MagicBar.LoadBitmap(SasukeMagicBar, RGB(0, 0, 0));
		bloodbackground.LoadBitmap(blood_background);
		Blood.LoadBitmap(blood);
		Blood2.LoadBitmap(blood_2);
		Magic.LoadBitmap(Magic1);
		Chakura.LoadBitmap(chakura_black);
		SkillText.AddBitmap(SasukeSkillText1, RGB(255, 255, 255));
		SkillText.AddBitmap(SasukeSkillText2, RGB(255, 255, 255));
		SkillText.AddBitmap(SasukeSkillText3, RGB(255, 255, 255));
		SkillText.AddBitmap(SasukeSkillText4, RGB(255, 255, 255));
		SkillText.SetDelayCount(1);
		SkillText.Reset();
		SetXY(_x, _y);
		numOfMagic = 0;
		numOfChakura = 64;
		numOfBlood = _numOfBlood;
		characteristic = _characteristic;
		characterUI.SetTopLeft(_x, 10);
		MagicBar.SetTopLeft(84, 74);
		if(_characteristic == "AI") bloodbackground.SetTopLeft(_x+3, 29);
		else bloodbackground.SetTopLeft(84, 29);
		/////////////////////////////////////////////////////////////////////////////////////
	}
	int bloodcount::GetX1()
	{
		return x;
	}
	int bloodcount::GetY1()
	{
		return y;
	}
	void bloodcount::SetXY(int _x, int _y)
	{
```

```
	x = _x;
	y = _y;
}
void bloodcount::ChangeBlood(int _numOfBlood)
{
	numOfBlood = _numOfBlood;
}
void bloodcount::SetisMolding(bool flag,int _numOfMagic)
{
	isMolding = flag;
	numOfMagic = _numOfMagic;
}
void bloodcount::ChangeChakura(int _chakura)
{
	numOfChakura = _chakura;
}
int bloodcount::GetChakura()
{
	return numOfChakura;
}
void bloodcount::OnShow()
{
	characterUI.ShowBitmap(2);
	bloodbackground.ShowBitmap();
	//////////////////////消耗越多查克拉越少//////////////////////////
	for (int i = 0; i < 64 - numOfChakura; i++)
	{
		Chakura.SetTopLeft(84 + (64 - i)*2, 48);
		Chakura.ShowBitmap();
	}
	//////////////////////血量減少//////////////////////////////////
	if (characteristic == "player")
	{
		for (int i = 0; i < numOfBlood && i < 65; i++)
		{
			Blood2.SetTopLeft(84 + (i * 2), 29);
			Blood2.ShowBitmap();
		}
		if (numOfBlood >= 64)
		{
			for (int i = 0; i < numOfBlood - 63; i++)
			{
				Blood.SetTopLeft(84 + (i * 2), 29);
				Blood.ShowBitmap();
			}
		}
		//////////////////////集氣增加//////////////////////////
		if (isMolding && numOfMagic>0)
		{
			MagicBar.ShowBitmap(2);
			int num = numOfMagic;
			if (numOfMagic > 128 && numOfMagic < 512) num -= (numOfMagic / 128) * 128;
			for (int i = 0; i < 128-num; i++)
			{
				Magic.SetTopLeft(84 + (127-i), 77);
				Magic.ShowBitmap();
			}
			SkillText.SetTopLeft(84, 97);
			SkillText.Movetonum((numOfMagic / 128));
			SkillText.OnShow();
		}
	}
	else
	{
		//////////////////////////血量減少//////////////////////////
		for (int i = 0; i < numOfBlood && i < 65; i++)
		{
			Blood2.SetTopLeft(x + 3 + ((64-i) * 2), 29);
			Blood2.ShowBitmap();
		}
		if (numOfBlood >= 64)
		{
			for (int i = 0; i < numOfBlood - 63; i++)
			{
				Blood.SetTopLeft(x + 3 + ((64-i) * 2), 29);
				Blood.ShowBitmap();
```

```
                        }
                    }
                    ////////////////////////////////////////////////////////////
                }
        }
        int bloodcount::GetBlood()
        {
                return numOfBlood;
        }
}
```

EnemyAI.h

```
#ifndef ENEMYAI_H
#define ENEMYAI_H
#include "Shurikan.h"
#include "Sasuke.h"
#include "SkillStart.h"
namespace game_framework {
        class EnemyAI
        {
        public:
                EnemyAI();
                virtual ~EnemyAI();
                virtual int     GetX1();                                // 角色左上角 x 座標
                virtual int     GetY1();                                // 角色左上角 y 座標
                virtual int     GetX2();                                // 角色右下角 x 座標
                virtual int     GetY2();                                // 角色右下角 y 座標
                virtual void Initialize()=0;                    // 設定角色為初始值
                virtual void LoadBitmap()=0;                    // 載入圖形
                virtual void UI(bool Hit) = 0;
                void SetonFloor();
                void OnMove();
                void jump();                                    // 跳躍
                virtual void OnShow()=0;                                // 將角色圖形貼到畫面
                void SetMovingLeft(bool flag);      // 設定是否正在往左移動
                void SetMovingRight(bool flag); // 設定是否正在往右移動
                void SetMovingUp(bool flag);        // 設定是否正在往上移動
                void SetDirection(int flag);
                void SetAttacked(int flag);
                void SetDie(int direction);
                void SetSkillsuccess();
                bool GetSkillsuccess();
                int GetDirection();
                bool GetMovingLeft();
                bool GetMovingRight();
                void SetXY(int nx, int ny);             // 設定角色左上角座標
                void SetisShow(bool flag);
                bool GetMove();
                void Addattacknum();
                bool GetAttack();
                bool GetDie();
                bool GetAttacked();
                int Getblood();
                int Setblood(int _blood);
                int GetAttacknum();
                int GetCurrentBitmapNumber();
                void SetSasukeX(int x);
                void SetMove(bool flag);
                vector <Shurikan*> GetShurikan();
                CAnimation GetSkillMovie();
                CAnimation* GetSkillsuccessMovie();
                void SetShurikanDelete(int flag);
                virtual bool GetisSkill();
                void SetPlayer(Sasuke *_player);
                skillstart* Getskillstart();
                void Attack();
                void Attacked();
                void Throw();
                void MovePlayer();
                void Die();
                void Stand();
                void SetSkill();
                void SetSkill2();
        protected:
                int floor;
                int velocity;                   // 目前的速度(點/次)
                int initial_velocity;    // 初始速度
```

```
		bool rising;			// true 表上升、false 表下降
		int x, y;				// 角色左上角座標
		int SasukeX;
		bool isMovingLeft;		// 是否正在往左移動
		bool isMovingRight;		// 是否正在往右移動
		bool isMovingUp;		// 是否正在往上移動
		bool isDie;
		bool isSkill,Skillsuccess;
		int isAttacked;
		bool isAttack;
		bool isShooting;
		bool isThrow;
		int Playerdirection;
		int direction;
		int times;
		int Skill;
		int step_size;
		int Moldingdelay;
		bool Move;
		bool isShow;
		int StandHeight, StandWidth;
		int StandStart,StandNum,RunStart,RunNum,JumpStart,JumpNum,AttackStart,AttackNum, Attack2Start, Attack2Num,
		Attack3Start,Attack3Num,ThrowStart,ThrowNum,HurtStart,HurtNum,DieStart,DieNum,SkillStart,Skillnum,Total;
		CAnimation animation;		// 利用動畫作圖形
		CAnimation SkillMovie;
		CAnimation SkillsuccessMovie;
		skillstart Movie;
		CMovingBitmap MagicBar;
		CMovingBitmap Black;
		int MagicX, MagicY;
		vector <Shurikan*> shoot;
		int attacknum;
		int distance;
		bool Wall;
		int Diedelay;
		int Blood;
		int delay;
		Sasuke* Player;
		bool Movieflag;
	};
}
#endif
```

EnemyAI.cpp

```
#include "stdafx.h"
#include "Resource.h"
#include <mmsystem.h>
#include <ddraw.h>
#include "audio.h"
#include "gamelib.h"
#include "EnemyAI.h"

namespace game_framework {
	EnemyAI::EnemyAI()
	{
		unsigned seed;
		seed = (unsigned)time(NULL); // 取得時
		srand(seed); // 以時間序列當亂數種子
		times = 0;
		times = rand() % 75;
		Diedelay = 30;
	}
	EnemyAI::~EnemyAI()
	{
		int num = (int)shoot.size();
		for (int i = 0; i < num; i++)
		{
			delete shoot[0];
			shoot.erase(shoot.begin());
		}
	}
	int EnemyAI::GetX1()
	{
		//////////////////////////////////////////////////////////圖片 x 座標修正//////////////////////////////////////////////
		if (direction == 1) return x;
		return x + (StandWidth - animation.Width()) * 2;
		//////////////////////////////////////////////////////////////////////////////////////////////////////////////////
```

```
}
int EnemyAI::GetY1()
{
	//////////////////////////////////////////////////////圖片 y 座標修正//////////////////////////////////////////
	return y + (StandHeight - animation.Height()) * 2;
	//////////////////////////////////////////////////////////////////////////////////////////////////////////////
}
int EnemyAI::GetX2()
{
	return GetX1() + animation.Width() * 2;
}
int EnemyAI::GetY2()
{
	return GetY1() + animation.Height() * 2;
}
void EnemyAI::SetonFloor()
{
	y = floor - StandHeight * 2;
}
void EnemyAI::OnMove()
{
	animation.SetDelayCount(1);
	//////////////被攻擊//////////////////////
	if (isAttacked && !isDie && !isSkill) Attacked();
	else if (Move)
	{
		//////////////////////使用技能、攻擊、集氣、移動、跳躍、恢復查克拉//////////////////////////////////////
		if (attacknum > 0) Attack();
		else if (isThrow&&GetY2() >= floor) Throw();
		else if (isMovingLeft || isMovingRight || isMovingUp) MovePlayer();
		else Stand();
		if(!Player->GetSkillsuccess())jump();
		//////////////////////////////////////////////////////////////////////////////////////////////
	}
	//////////////////////////////倒地/////////////////////////////////////
	if (isDie && !isSkill) Die();
	//////////////////////////投擲物移動/////////////////////////////////
	if (shoot.size() != 0)
	{
		for (int count = 0; count < (int)shoot.size(); count++)
		{
			shoot[count]->Move();
			if ((shoot[count]->GetX2() < 0 && direction==0) || (shoot[count]->GetX1() > 640 && direction == 1))
			{
				delete shoot[count];
				shoot.erase(shoot.begin() + count);
			}
		}
	}
	//////////////////////////////////////////////////////////////////////////
}
void EnemyAI::jump()
{
	/////////////////////////////////////////跳躍/////////////////////////////////////
	if (rising)                                  // 上升狀態
	{
		if (velocity > 0)
		{
			y -= velocity;  // 當速度 > 0 時，y 軸上升(移動 velocity 個點，velocity 的單位為 點/次)
			velocity -= 2;
		}
		else
		{
			rising = false; // 當速度 <= 0，上升終止，下次改為下降
			velocity = 1;   // 下降的初速(velocity)為 1
		}
	}
	else                                 // 下降狀態
	{
		if (GetY2() < floor)   // 當 y 座標還沒碰到地板
		{
			y += velocity; // y 軸下降(移動 velocity 個點，velocity 的單位為 點/次)
			velocity += 2;
			if (attacknum == 0) animation.Movetonum(JumpStart + direction * Total);
		}
		else
```

```
		{
			y = floor - StandHeight*2;	// 當 y 座標低於地板，更正為地板上
			velocity = initial_velocity; // 重設上升初始速度
			if (isMovingUp && attacknum == 0)
			{
				Move = false;
				//////////////////////跳躍結束動作//////////////////////////
				if (!(animation.GetCurrentBitmapNumber() >= JumpStart+2 + direction * Total &&
				animation.GetCurrentBitmapNumber() <= JumpNum + JumpStart + direction * Total))
				animation.Movetonum(JumpStart+2 + direction * Total);
				else animation.OnMove();
				if (animation.GetCurrentBitmapNumber() == JumpNum + JumpStart +1 + direction * Total)
				{
					Move = true;
					isMovingUp = false;
				}
				////////////////////////////////////////////////////
			}
		}
	}
}
void EnemyAI::SetMovingLeft(bool flag)
{
	isMovingLeft = flag;
}
void EnemyAI::SetMovingRight(bool flag)
{
	isMovingRight = flag;
}
void EnemyAI::SetMovingUp(bool flag)
{
	isMovingUp = flag;
}
void EnemyAI::SetDirection(int flag)
{
	direction = flag;
}
void EnemyAI::SetAttacked(int flag)
{
	if(!isDie) isAttacked = flag;
}
void EnemyAI::SetDie(int _direction)
{
	direction = _direction;
	CAudio::Instance()->Play(29);
	isDie = true;
}
void EnemyAI::SetSkillsuccess()
{
	Skillsuccess = true;
}
bool EnemyAI::GetSkillsuccess()
{
	return Skillsuccess;
}
int EnemyAI::GetDirection()
{
	return direction;
}
bool EnemyAI::GetMovingLeft()
{
	return isMovingLeft;
}
bool EnemyAI::GetMovingRight()
{
	return isMovingRight;
}
void EnemyAI::SetXY(int nx, int ny)
{
	x = nx, y = ny;
}
void EnemyAI::SetisShow(bool flag)
{
	isShow = flag;
}
bool EnemyAI::GetMove()
```

```
{
	return Move;
}
void EnemyAI::Addattacknum()
{
	attacknum++;
}
bool EnemyAI::GetAttack()
{
	return isAttack;
}
bool EnemyAI::GetDie()
{
	return isDie;
}
bool EnemyAI::GetAttacked()
{
	return isAttacked;
}
int EnemyAI::Getblood()
{
	return Blood;
}
int EnemyAI::Setblood(int _blood)
{
	return Blood = _blood;
}
int EnemyAI::GetAttacknum()
{
	return attacknum;
}
int EnemyAI::GetCurrentBitmapNumber()
{
	return animation.GetCurrentBitmapNumber();
}
void EnemyAI::SetSasukeX(int x)
{
	SasukeX = x;
}
void EnemyAI::SetMove(bool flag)
{
	Move = flag;
}
vector<Shurikan*> EnemyAI::GetShurikan()
{
	return shoot;
}
CAnimation EnemyAI::GetSkillMovie()
{
	return SkillMovie;
}
CAnimation * EnemyAI::GetSkillsuccessMovie()
{
	return &SkillsuccessMovie;
}
void EnemyAI::SetShurikanDelete(int flag)
{
	delete shoot[flag];
	shoot.erase(shoot.begin() + flag);
}
bool EnemyAI::GetisSkill()
{
	return isSkill;
}
void EnemyAI::SetPlayer(Sasuke *_player)
{
	Player = _player;
}
skillstart* EnemyAI::Getskillstart()
{
	return &Movie;
}
void EnemyAI::Attack()
{
	//////////////////////////////////////////////攻擊動畫//////////////////////////////////////////////////////////
	if (!(animation.GetCurrentBitmapNumber() >= AttackStart + direction * Total && animation.GetCurrentBitmapNumber() < AttackStart
```

```
            + AttackNum + direction * Total))
                animation.Movetonum(AttackStart + direction * Total);
            else
                animation.OnMove();
            if (animation.GetCurrentBitmapNumber() == AttackStart + AttackNum + direction * Total)
            {
                attacknum = 0;
                Skill = rand() % 10;
                isAttack = false;
            }
            ////////////////////////////////////////////////////////////////////////////////////////////////////////////////////
        }
        void EnemyAI::Attacked()
        {
            ////////////////////////////////////////////////////////////被攻擊////////////////////////////////////////////////////
            if (!(animation.GetCurrentBitmapNumber() >= HurtStart + direction * Total && animation.GetCurrentBitmapNumber() < HurtStart +
            HurtNum + direction * Total))
            {
                animation.Movetonum(HurtStart + direction * Total);
                CAudio::Instance()->Play(29);
            }
            else
                animation.OnMove();
            ////////////////////////////////////////////////////角色擊退////////////////////////////////////////////////////
            if (direction == 0 && GetX2() + 4 <= 640) x += 4;
            else if (direction == 1 && GetX1() - 4 >= 0)x -= 4;
            ////////////////////////////////////////////////////減少血量////////////////////////////////////////////////////
            if (animation.GetCurrentBitmapNumber() == HurtStart + HurtNum + direction * Total) isAttacked = false;
            if (Blood > 0) Blood -= 2;
            attacknum = 0;
            isAttack = false;
            ////////////////////////////////////////////////////////////////////////////////////////////////////////////////////
        }
        void EnemyAI::Throw()
        {
            ////////////////////////////////////////////////////////丟擲物品////////////////////////////////////////////////////
            if (!(animation.GetCurrentBitmapNumber() >= ThrowStart + direction * Total && animation.GetCurrentBitmapNumber() < ThrowStart +
ThrowNum + direction * Total))
                animation.Movetonum(ThrowStart + direction * Total);
            else
                animation.OnMove();
            if (animation.GetCurrentBitmapNumber() == ThrowStart + ThrowNum + direction * Total)
            {
                shoot.push_back(new Shurikan(this->GetX1(), (this->GetY2() - this->GetY1()) / 3 + this->GetY1(), direction));//新增一個 Shurikan
                isThrow = false;
                Skill = rand() % 10;//重新選擇一項攻擊方式
            }
            ////////////////////////////////////////////////////////////////////////////////////////////////////////////////////
        }
        void EnemyAI::MovePlayer()
        {
            ////////////////////////////////////////////////////左移////////////////////////////////////////////////////
            if (isMovingLeft)
            {
                if (GetX1() > 0)
                    x -= step_size;
                if (GetY2() >= floor)
                {
                    if (!(animation.GetCurrentBitmapNumber() >= RunStart + direction * Total && animation.GetCurrentBitmapNumber() <
                    RunStart + RunNum + direction * Total))
                    {
                        animation.Movetonum(RunStart + direction * Total);
                    }
                    else
                    {
                        animation.OnMove();
                    }
                }
                direction = 0;
            }
            ////////////////////////////////////////////////////右移////////////////////////////////////////////////////
            else if (isMovingRight)
            {
                if (GetX2() < 640)
                    x += step_size;
                if (GetY2() >= floor)
```

```
                {
                        if (!(animation.GetCurrentBitmapNumber() >= RunStart + direction * Total && animation.GetCurrentBitmapNumber() <
                        RunStart + RunNum + direction * Total))
                        {
                                animation.Movetonum(RunStart + direction * Total);
                        }
                        else
                        {
                                animation.OnMove();
                        }
                }
                direction = 1;
        }
}
void EnemyAI::Die()
{
        ////////////////////////////////////人物倒地////////////////////////////////////////////////////////////
        Move = false;
        if (animation.GetCurrentBitmapNumber() < DieStart + direction * Total || animation.GetCurrentBitmapNumber() > DieStart + DieNum +
        direction * Total)      animation.Movetonum(DieStart + direction * Total);
        if (animation.GetCurrentBitmapNumber() >= DieStart + 12 + direction * Total && animation.GetCurrentBitmapNumber() <= DieStart + 17 +
        direction * Total)
        {
                velocity -= 3;
                if (GetY2() < floor) y -= velocity;//  當後半段動畫角色從空中落下
        }
        if (animation.GetCurrentBitmapNumber() >= DieStart + direction * Total && animation.GetCurrentBitmapNumber() <= DieStart + 5 +
        direction * Total)
        {
                y -= velocity;//  當前半段動畫角色被擊飛至空中
                velocity -= 3;
        }
        if (animation.GetCurrentBitmapNumber() >= DieStart + direction * Total && animation.GetCurrentBitmapNumber() <= DieStart + 17 +
        direction * Total)
        {
                ////////////////////////////////////角色往後飛////////////////////////////////
                if (direction == 0 && GetX2() + 7 <= 640) x += 7;
                else if (direction == 1 && GetX1() - 7 >= 0) x -= 7;
                ////////////////////////////////////////////////////////////////////////////////
        }
        if (GetY2() < floor)    y -= velocity;//////////////////確保不超過地面且到達地面
        if (animation.GetCurrentBitmapNumber() == DieStart + 18 + direction * Total) CAudio::Instance()->Play(30);
        if (animation.GetCurrentBitmapNumber() == DieStart + DieNum + direction * Total)
        {
                ////////////////////////////////倒地延遲////////////////////////////////
                if (Diedelay-- <= 0)
                {
                        isDie = false;
                        Move = true;
                        isAttacked = false;
                        SkillMovie.Reset();
                        Diedelay = 30;
                        velocity = initial_velocity;
                        y = floor - StandHeight * 2;
                        attacknum = 0;
                }
                ////////////////////////////////////////////////////////////////////////
        }
        else animation.OnMove();
        ////////////////////////////////減少血量////////////////////////////////////
        if (Blood > 0 && delay-- <= 0 && animation.GetCurrentBitmapNumber() != DieStart + DieNum + direction * Total)
        {
                Blood -= 1;
                delay = 2;
        }
        ////////////////////////////////////////////////////////////////////////////
}
void EnemyAI::Stand()
{
        ////////////////////////////////站在原地////////////////////////////////////////
        if (!(animation.GetCurrentBitmapNumber() >= StandStart + direction * Total && animation.GetCurrentBitmapNumber() < StandStart +
StandNum + direction * Total))
                animation.Movetonum(StandStart + direction * Total);
        else
                animation.OnMove();
        ////////////////////////////////////////////////////////////////////////////////
```

```
	}
	void EnemyAI::SetSkill()
	{
		Skill = 9;//demo 用
	}
	void EnemyAI::SetSkill2()
	{
		Skill = 8;//demo 用
	}
}
```

Neji.h

```
#ifndef NEJI_H
#define NEJI_H
#include "EnemyAI.h"
namespace game_framework {
	class Neji : public EnemyAI
	{
	public:
		Neji();
		~Neji();
		int	GetX1();
		int	GetY1();
		int	GetX2();
		int	GetY2();
		void AttackMovie();
		void RunSkillsuccessMovie();
		void Initialize();
		void LoadBitmap();
		void UI(bool Hit);
		void SetpicXY();
		void OnShow();
	private:
		int picx, picy;
		int Skilldelay;
		int SasukeDistance;
		int _direction, _distance;
	};
}
#endif
```

Neji.cpp

```
#include "stdafx.h"
#include "Resource.h"
#include <mmsystem.h>
#include <ddraw.h>
#include "audio.h"
#include "gamelib.h"
#include "Neji.h"
namespace game_framework {
	Neji::Neji()
	{
		Initialize();
	}
	Neji::~Neji()
	{
	}
	int Neji::GetX1()
	{
		//////////////////////////////////////////////////////////圖片 x 座標修正//////////////////////////////////////////////
		if (isSkill && SkillMovie.GetCurrentBitmapNumber() >= 18 && SkillMovie.GetCurrentBitmapNumber() <= 38) return x + (StandWidth -
		SkillMovie.Width()) * 2 + 126;
		if (isSkill && SkillMovie.GetCurrentBitmapNumber() >= 57 && SkillMovie.GetCurrentBitmapNumber() <= 77) return x + (StandWidth -
		SkillMovie.Width()) * 2 + 160;
		if (isSkill && direction == 0) return x + (StandWidth - SkillMovie.Width()) * 2;
		if (direction == 1) return x;
		return x + (StandWidth - animation.Width()) * 2;
		//////////////////////////////////////////////////////////////////////////////////////////////////////////////////////
	}
	int Neji::GetY1()
	{
		//////////////////////////////////////////////////////////圖片 y 座標修正//////////////////////////////////////////////
		if(isSkill && SkillMovie.GetCurrentBitmapNumber() >=18+direction*39 && SkillMovie.GetCurrentBitmapNumber()<=38+ direction *39)
		return y + (StandHeight - SkillMovie.Height()) * 2 + 64;
		if (isSkill) return y + (StandHeight - SkillMovie.Height()) * 2 + 30;
		return y + (StandHeight - animation.Height()) * 2;
		//////////////////////////////////////////////////////////////////////////////////////////////////////////////////////
```

```
}
int Neji::GetX2()
{
	if (isSkill) return GetX1() + SkillMovie.Width() * 2;
	return GetX1() + animation.Width() * 2;
}
int Neji::GetY2()
{
	if (isSkill) return GetY1() + SkillMovie.Height() * 2 - 30;
	return GetY1() + animation.Height() * 2;
}
void Neji::AttackMovie()
{
	//////////////////////////////////////////////////////////播放技能動畫//////////////////////////////////////////////
	if (Movie.isStart())
	{
		Movie.OnMove();
		Player->SetMove(false);
	}
	//////////////////////////////////////////////////////////技能判定//////////////////////////////////////////////////
	if (!Movie.isStart())
	{
		if (!Skillsuccess) { SkillMovie.OnMove(); Player->SetMove(true); }
		else RunSkillsuccessMovie();
		if (SkillMovie.GetCurrentBitmapNumber() == 38 + direction * 39)
		{
			Move = true;
			isSkill = false;
			Movieflag = false;
			isDie = false;
			isAttacked = false;
			SkillMovie.Reset();
			Skill = rand() % 10;//重新選擇攻擊方式
		}
	}
	//////////////////////////////////////////////////////////////////////////////////////////////////////////////////
}
void Neji::RunSkillsuccessMovie()
{
	//////////////////////////////////////////////////////////技能成功//////////////////////////////////////////////////
	Player->SetDirection(direction);
	if (SkillsuccessMovie.GetCurrentBitmapNumber() == 11 + direction * 57) { Player->SetAttacked(true); }
	else if (SkillsuccessMovie.GetCurrentBitmapNumber() == 17 + direction * 57) { Player->SetAttacked(true); }
	else if (SkillsuccessMovie.GetCurrentBitmapNumber() == 32 + direction * 57) { Player->SetAttacked(true); }
	else if (SkillsuccessMovie.GetCurrentBitmapNumber() == 44 + direction * 57) { Player->SetAttacked(true); }
	else if (SkillsuccessMovie.GetCurrentBitmapNumber() == 50 + direction * 57) { Player->SetAttacked(true); }
	else if (SkillsuccessMovie.GetCurrentBitmapNumber() == 53 + direction * 57) { Player->SetAttacked(true); }
	if (SkillsuccessMovie.GetCurrentBitmapNumber() == 56 + direction * 57)
	{
		Skillsuccess = false;
		isSkill = false;
		Move = true;
		Movieflag = false;
		isDie = false;
		isAttacked = false;
		Skilldelay = 33;
		SkillsuccessMovie.Reset();
		SkillMovie.Reset();
		Player->SetDie();
		Skill = rand() % 10;//重新選擇攻擊方式
		isShow = true;
	}
	else SkillsuccessMovie.OnMove();
	SkillsuccessMovie.SetTopLeft(picx, picy - SkillsuccessMovie.Height() * 2);
	if(direction==0)picx -= 1;
	else picx += 1;
	//////////////////////////////////////////////////////////////////////////////////////////////////////////////////
}
void Neji::Initialize()
{
	//////////////////////////////////////////////////初始化//////////////////////////////////////////////////////////
	floor = 460;
	const int X_POS = 500;
	const int Y_POS = 460 - 39;
	const int INITIAL_VELOCITY = 18;
	initial_velocity = INITIAL_VELOCITY;
```

```
	velocity = initial_velocity;
	x = X_POS;
	y = Y_POS;
	isThrow=Movieflag = Skillsuccess = isSkill = Wall = isAttack = isDie = isShooting = isAttacked = rising = isMovingLeft = isMovingRight =
	isMovingUp = false;
	step_size = 10;
	isShow = Move = true;
	attacknum = 0;
	direction = 0;
	StandHeight = 35;
	StandWidth = 28;
	Total = 109, StandStart = 0, StandNum = 2, RunStart = 3, RunNum = 17, JumpStart = 21, JumpNum = 0, AttackStart = 22, AttackNum = 50,
	ThrowStart = 73, ThrowNum = -1, HurtStart = 73, HurtNum = 2, DieStart = 79, DieNum = 29;
	distance = 200;
	Playerdirection = 1;
	Blood = 128;
	delay = 2;
	Skilldelay = 33;
	Skill = rand() % 10;
	Diedelay = 30;
	////////////////////////////////////////////////////////////////////////////////////////////////////////////////
}
void Neji::UI(bool Hit)
{
	//////////////////////////////////////////////依照對方位置決定面向哪裡//////////////////////////////////////////////
	if ((Player->GetX2()-Player->GetX1())/2+Player->GetX1() < (GetX2()-GetX1())/2+GetX1())
	{
		if (!GetAttack() && GetAttacked()==0 && !GetDie() && !GetisSkill() && GetAttacknum() == 0)
		{
			_direction = 1;
			SetDirection(0);
		}
	}
	else if ((Player->GetX2() - Player->GetX1()) / 2 + Player->GetX1() > (GetX2() - GetX1()) / 2 + GetX2())
	{
		if (!GetAttack() && GetAttacked()==0 && !GetDie() && !GetisSkill() && GetAttacknum() == 0)
		{
			_direction = 0;
			SetDirection(1);
		}
	}
	//////////////////////////////////////////////判斷距離//////////////////////////////////////////////
	if (!isAttack) Playerdirection = _direction;
	if (Playerdirection == 0) _distance = Player->GetX1() - GetX2();
	else _distance = Player->GetX2() - GetX1();
	_distance = abs(_distance);
	distance = _distance;
	//////////////////////////////////////////////擊倒後使用回天//////////////////////////////////////////////
	if (!Player->GetSkillsuccess())
	{
		if (Diedelay <= 1 || (isSkill && (SkillMovie.GetCurrentBitmapNumber() >= 0 + direction * 39 &&
		SkillMovie.GetCurrentBitmapNumber() <= 17 + direction * 39 || SkillMovie.GetCurrentBitmapNumber() == 0)))
		{
			if (SkillMovie.GetCurrentBitmapNumber() < 39 && direction == 1)
				SkillMovie.Movetonum(39);
			isSkill = true;
			Move = false;
			isAttacked = false;
			isDie = false;
			SkillMovie.OnMove();
			if (SkillMovie.GetCurrentBitmapNumber() == 17|| SkillMovie.GetCurrentBitmapNumber() == 17 + 39)
			{
				isDie = false;
				Move = true;
				isAttacked = false;
				SkillMovie.Reset();
				Diedelay = 30;
				velocity = initial_velocity;
				y = floor - StandHeight * 2;
				isSkill = false;
				attacknum = 0;
			}
		}
		//////////////////////////////////////////////技能施放//////////////////////////////////////////////
		else if (Skill == 9 && times <= 0 && Move && _distance < 190 || isSkill)
		{
```

```
                        if (!Movieflag)
                        {
                                if (Player->GetX1() < GetX2()) direction = 0;
                                else if (Player->GetX2() > GetX1()) direction = 1;
                                SkillMovie.Movetonum(18 + direction * 39);
                                Movieflag = true;
                                Movie.Init();
                                CAudio::Instance()->Play(31);
                        }
                        isMovingRight = false;
                        isMovingLeft = false;
                        isSkill = true;
                        Move = false;
                        isAttacked = false;
                        AttackMovie();
                        times = rand() % 75 + 30;
                }
                ////////////////////////////////////////////////////攻擊及移動判定/////////////////////////////////////////////
                else if (times <= 0 && Move)
                {
                        if (distance > 25)
                        {
                                if (Playerdirection == 1)
                                {
                                        isMovingRight = false;
                                        isMovingLeft = true;
                                }
                                else
                                {
                                        isMovingLeft = false;
                                        isMovingRight = true;
                                }
                        }
                        if (distance <= 25)
                        {
                                isAttack = true;
                                isMovingRight = false;
                                isMovingLeft = false;
                                attacknum = 1;
                                times = rand() % 30;
                                Wall = false;
                        }
                }
                ////////////////////////////////////////////////////攻擊擊中判定/////////////////////////////////////////////
                if (!Player->GetisMolding() && !Player->GetisSkill() && Hit && !Player->GetisDie() && (GetCurrentBitmapNumber() == 22 + 5 +
                GetDirection() * 109 || GetCurrentBitmapNumber() == 45 + GetDirection() * 109|| GetCurrentBitmapNumber() == 59  + GetDirection()
                * 109))
                {
                        Player->SetAttacked(true);
                        Player->ChangeChakura(Player->GetChakura() - 30);
                }
                ////////////////////////////////////////////////////技能擊中判定/////////////////////////////////////////////
                if (!Player->GetisMolding() && !Player->GetisSkill() && Hit && !Player->GetisDie() && GetisSkill() &&
                GetSkillMovie().GetCurrentBitmapNumber() >= 12 + GetDirection() * 39 && GetSkillMovie().GetCurrentBitmapNumber() <= 17 +
                GetDirection() * 39)
                {
                        Player->SetAttacked(true);
                        Player->SetDirection(direction);
                        Player->ChangeChakura(Player->GetChakura() - 30);
                        if (GetSkillMovie().GetCurrentBitmapNumber() == 16 + GetDirection() * 39) Player->SetDie();
                }
                ////////////////////////////////////////////////////技能動畫判定/////////////////////////////////////////////
                if (!Player->GetisMolding() && !Player->GetisSkill() && GetisSkill() && Hit && GetSkillMovie().GetCurrentBitmapNumber() >= 27
                + GetDirection() * 39 && GetSkillMovie().GetCurrentBitmapNumber() <= 38 + GetDirection() * 39 && !GetSkillsuccess())
                {
                        SetpicXY();
                        Player->SetMove(false);
                        SetisShow(false);
                        SetSkillsuccess();
                        GetSkillsuccessMovie()->Movetonum(0 + GetDirection() * 57);
                }
                ////////////////////////////////////////////////////角色 delay/////////////////////////////////////////////
                if (Move) times--;
        }
}
void Neji::SetpicXY()
```

```
{
	//////////////////////////////////////////////設定技能動畫位置到玩家旁邊//////////////////////////////////////////////
	if (direction == 0) picx = Player->GetX2() + 20;
	else picx = Player->GetX1() - 20 - StandWidth;
	picy = Player->GetY2();
}
void Neji::OnShow()
{
	//////////////////////////////////////////////顯示//////////////////////////////////////////////////////////
	for (int count = 0; count < (int)shoot.size(); count++) shoot[count]->OnShow();
	if (!isSkill)
	{
		animation.SetTopLeft(GetX1(), GetY1());
		animation.OnShow(2);
	}
	else if (!Skillsuccess)
	{
		SkillMovie.SetTopLeft(GetX1(), GetY1() - 30);
		SkillMovie.OnShow(2);
	}
	if (Skillsuccess) SkillsuccessMovie.OnShow(2);
}
void Neji::LoadBitmap()
{
	/////////////////////////////////////////////////////////////讀取圖片//////////////////////////////////////
	animation.AddBitmap(NejiStand, RGB(0, 128, 128));		//0
	animation.AddBitmap(NejiStand, RGB(0, 128, 128));		//1
	animation.AddBitmap(NejiStand, RGB(0, 128, 128));		//2
	animation.AddBitmap(NejiRun1, RGB(0, 128, 128));		//3
	animation.AddBitmap(NejiRun1, RGB(0, 128, 128));		//4
	animation.AddBitmap(NejiRun1, RGB(0, 128, 128));		//5
	animation.AddBitmap(NejiRun2, RGB(0, 128, 128));		//6
	animation.AddBitmap(NejiRun2, RGB(0, 128, 128));		//7
	animation.AddBitmap(NejiRun2, RGB(0, 128, 128));		//8
	animation.AddBitmap(NejiRun3, RGB(0, 128, 128));		//9
	animation.AddBitmap(NejiRun3, RGB(0, 128, 128));		//10
	animation.AddBitmap(NejiRun3, RGB(0, 128, 128));		//1
	animation.AddBitmap(NejiRun4, RGB(0, 128, 128));		//2
	animation.AddBitmap(NejiRun4, RGB(0, 128, 128));		//3
	animation.AddBitmap(NejiRun4, RGB(0, 128, 128));		//4
	animation.AddBitmap(NejiRun5, RGB(0, 128, 128));		//5
	animation.AddBitmap(NejiRun5, RGB(0, 128, 128));		//6
	animation.AddBitmap(NejiRun5, RGB(0, 128, 128));		//7
	animation.AddBitmap(NejiRun6, RGB(0, 128, 128));		//8
	animation.AddBitmap(NejiRun6, RGB(0, 128, 128));		//9
	animation.AddBitmap(NejiRun6, RGB(0, 128, 128));		//20
	animation.AddBitmap(NejiSkill15, RGB(0, 128, 128));		//1
	animation.AddBitmap(NejiCombo1, RGB(0, 128, 128));		//2
	animation.AddBitmap(NejiCombo1, RGB(0, 128, 128));		//3
	animation.AddBitmap(NejiCombo1, RGB(0, 128, 128));		//4
	animation.AddBitmap(NejiCombo2, RGB(0, 128, 128));		//5
	animation.AddBitmap(NejiCombo2, RGB(0, 128, 128));		//6
	animation.AddBitmap(NejiCombo2, RGB(0, 128, 128));		//7
	animation.AddBitmap(NejiCombo3, RGB(0, 128, 128));		//8
	animation.AddBitmap(NejiCombo3, RGB(0, 128, 128));		//9
	animation.AddBitmap(NejiCombo3, RGB(0, 128, 128));		//30
	animation.AddBitmap(NejiCombo4, RGB(0, 128, 128));		//1
	animation.AddBitmap(NejiCombo4, RGB(0, 128, 128));		//2
	animation.AddBitmap(NejiCombo4, RGB(0, 128, 128));		//3
	animation.AddBitmap(NejiCombo5, RGB(0, 128, 128));		//4
	animation.AddBitmap(NejiCombo5, RGB(0, 128, 128));		//5
	animation.AddBitmap(NejiCombo5, RGB(0, 128, 128));		//6
	animation.AddBitmap(NejiCombo6, RGB(0, 128, 128));		//7
	animation.AddBitmap(NejiCombo6, RGB(0, 128, 128));		//8
	animation.AddBitmap(NejiCombo6, RGB(0, 128, 128));		//9
	animation.AddBitmap(NejiCombo7, RGB(0, 128, 128));		//40
	animation.AddBitmap(NejiCombo7, RGB(0, 128, 128));		//1
	animation.AddBitmap(NejiCombo7, RGB(0, 128, 128));		//2
	animation.AddBitmap(NejiCombo8, RGB(0, 128, 128));		//3
	animation.AddBitmap(NejiCombo8, RGB(0, 128, 128));		//4
	animation.AddBitmap(NejiCombo8, RGB(0, 128, 128));		//5
	animation.AddBitmap(NejiCombo9, RGB(0, 128, 128));		//6
	animation.AddBitmap(NejiCombo9, RGB(0, 128, 128));		//7
	animation.AddBitmap(NejiCombo9, RGB(0, 128, 128));		//8
	animation.AddBitmap(NejiCombo10, RGB(0, 128, 128));		//9
	animation.AddBitmap(NejiCombo10, RGB(0, 128, 128));		//50
```

```
animation.AddBitmap(NejiCombo10, RGB(0, 128, 128));		//1
animation.AddBitmap(NejiCombo11, RGB(0, 128, 128));		//2
animation.AddBitmap(NejiCombo11, RGB(0, 128, 128));		//3
animation.AddBitmap(NejiCombo11, RGB(0, 128, 128));		//4
animation.AddBitmap(NejiCombo12, RGB(0, 128, 128));		//5
animation.AddBitmap(NejiCombo12, RGB(0, 128, 128));		//6
animation.AddBitmap(NejiCombo12, RGB(0, 128, 128));		//7
animation.AddBitmap(NejiCombo13, RGB(0, 128, 128));		//8
animation.AddBitmap(NejiCombo13, RGB(0, 128, 128));		//9
animation.AddBitmap(NejiCombo13, RGB(0, 128, 128));		//60
animation.AddBitmap(NejiCombo14, RGB(0, 128, 128));		//1
animation.AddBitmap(NejiCombo14, RGB(0, 128, 128));		//2
animation.AddBitmap(NejiCombo14, RGB(0, 128, 128));		//3
animation.AddBitmap(NejiCombo15, RGB(0, 128, 128));		//4
animation.AddBitmap(NejiCombo15, RGB(0, 128, 128));		//5
animation.AddBitmap(NejiCombo15, RGB(0, 128, 128));		//6
animation.AddBitmap(NejiCombo16, RGB(0, 128, 128));		//7
animation.AddBitmap(NejiCombo16, RGB(0, 128, 128));		//8
animation.AddBitmap(NejiCombo16, RGB(0, 128, 128));		//9
animation.AddBitmap(NejiCombo17, RGB(0, 128, 128));		//70
animation.AddBitmap(NejiCombo17, RGB(0, 128, 128));		//1
animation.AddBitmap(NejiCombo17, RGB(0, 128, 128));		//2
animation.AddBitmap(NejiHurt1, RGB(0, 128, 128));		//3
animation.AddBitmap(NejiHurt1, RGB(0, 128, 128));		//4
animation.AddBitmap(NejiHurt1, RGB(0, 128, 128));		//5
animation.AddBitmap(NejiHurt2, RGB(0, 128, 128));		//6
animation.AddBitmap(NejiHurt2, RGB(0, 128, 128));		//7
animation.AddBitmap(NejiHurt2, RGB(0, 128, 128));		//8
animation.AddBitmap(NejiHurt1, RGB(0, 128, 128));		//9
animation.AddBitmap(NejiHurt1, RGB(0, 128, 128));		//80
animation.AddBitmap(NejiHurt1, RGB(0, 128, 128));		//1
animation.AddBitmap(NejiHurt1, RGB(0, 128, 128));		//2
animation.AddBitmap(NejiHurt1, RGB(0, 128, 128));		//3
animation.AddBitmap(NejiHurt2, RGB(0, 128, 128));		//4
animation.AddBitmap(NejiHurt2, RGB(0, 128, 128));		//5
animation.AddBitmap(NejiHurt2, RGB(0, 128, 128));		//6
animation.AddBitmap(NejiHurt2, RGB(0, 128, 128));		//7
animation.AddBitmap(NejiDie1, RGB(0, 128, 128));		//8
animation.AddBitmap(NejiDie1, RGB(0, 128, 128));		//9
animation.AddBitmap(NejiDie1, RGB(0, 128, 128));		//90
animation.AddBitmap(NejiDie1, RGB(0, 128, 128));		//1
animation.AddBitmap(NejiDie1, RGB(0, 128, 128));		//2
animation.AddBitmap(NejiDie1, RGB(0, 128, 128));		//3
animation.AddBitmap(NejiDie2, RGB(0, 128, 128));		//4
animation.AddBitmap(NejiDie2, RGB(0, 128, 128));		//5
animation.AddBitmap(NejiDie2, RGB(0, 128, 128));		//6
animation.AddBitmap(NejiDie3, RGB(0, 128, 128));		//7
animation.AddBitmap(NejiDie3, RGB(0, 128, 128));		//8
animation.AddBitmap(NejiDie3, RGB(0, 128, 128));		//9
animation.AddBitmap(NejiDie4, RGB(0, 128, 128));		//100
animation.AddBitmap(NejiDie4, RGB(0, 128, 128));		//1
animation.AddBitmap(NejiDie4, RGB(0, 128, 128));		//2
animation.AddBitmap(NejiDie4, RGB(0, 128, 128));		//3
animation.AddBitmap(NejiDie4, RGB(0, 128, 128));		//4
animation.AddBitmap(NejiDie4, RGB(0, 128, 128));		//5
animation.AddBitmap(NejiDie4, RGB(0, 128, 128));		//6
animation.AddBitmap(NejiDie4, RGB(0, 128, 128));		//7
animation.AddBitmap(NejiDie4, RGB(0, 128, 128));		//8
animation.AddBitmap(NejiStand2, RGB(0, 128, 128));		//9
animation.AddBitmap(NejiStand2, RGB(0, 128, 128));		//110
animation.AddBitmap(NejiStand2, RGB(0, 128, 128));		//1
animation.AddBitmap(NejiRun7, RGB(0, 128, 128));		//2
animation.AddBitmap(NejiRun7, RGB(0, 128, 128));		//3
animation.AddBitmap(NejiRun7, RGB(0, 128, 128));		//4
animation.AddBitmap(NejiRun8, RGB(0, 128, 128));		//5
animation.AddBitmap(NejiRun8, RGB(0, 128, 128));		//6
animation.AddBitmap(NejiRun8, RGB(0, 128, 128));		//7
animation.AddBitmap(NejiRun9, RGB(0, 128, 128));		//8
animation.AddBitmap(NejiRun9, RGB(0, 128, 128));		//9
animation.AddBitmap(NejiRun9, RGB(0, 128, 128));		//120
animation.AddBitmap(NejiRun10, RGB(0, 128, 128));		//1
animation.AddBitmap(NejiRun10, RGB(0, 128, 128));		//2
animation.AddBitmap(NejiRun10, RGB(0, 128, 128));		//3
animation.AddBitmap(NejiRun11, RGB(0, 128, 128));		//4
animation.AddBitmap(NejiRun11, RGB(0, 128, 128));		//5
animation.AddBitmap(NejiRun11, RGB(0, 128, 128));		//6
```

```
animation.AddBitmap(NejiRun12, RGB(0, 128, 128));          //7
animation.AddBitmap(NejiRun12, RGB(0, 128, 128));          //8
animation.AddBitmap(NejiRun12, RGB(0, 128, 128));          //9
animation.AddBitmap(NejiSkill44, RGB(0, 128, 128));        //130
animation.AddBitmap(NejiCombo18, RGB(0, 128, 128));        //1
animation.AddBitmap(NejiCombo18, RGB(0, 128, 128));        //2
animation.AddBitmap(NejiCombo18, RGB(0, 128, 128));        //3
animation.AddBitmap(NejiCombo19, RGB(0, 128, 128));        //4
animation.AddBitmap(NejiCombo19, RGB(0, 128, 128));        //5
animation.AddBitmap(NejiCombo19, RGB(0, 128, 128));        //6
animation.AddBitmap(NejiCombo21, RGB(0, 128, 128));        //7
animation.AddBitmap(NejiCombo21, RGB(0, 128, 128));        //8
animation.AddBitmap(NejiCombo21, RGB(0, 128, 128));        //9
animation.AddBitmap(NejiCombo22, RGB(0, 128, 128));        //140
animation.AddBitmap(NejiCombo22, RGB(0, 128, 128));        //1
animation.AddBitmap(NejiCombo22, RGB(0, 128, 128));        //2
animation.AddBitmap(NejiCombo23, RGB(0, 128, 128));        //3
animation.AddBitmap(NejiCombo23, RGB(0, 128, 128));        //4
animation.AddBitmap(NejiCombo23, RGB(0, 128, 128));        //5
animation.AddBitmap(NejiCombo24, RGB(0, 128, 128));        //6
animation.AddBitmap(NejiCombo24, RGB(0, 128, 128));        //7
animation.AddBitmap(NejiCombo24, RGB(0, 128, 128));        //8
animation.AddBitmap(NejiCombo25, RGB(0, 128, 128));        //9
animation.AddBitmap(NejiCombo25, RGB(0, 128, 128));        //150
animation.AddBitmap(NejiCombo25, RGB(0, 128, 128));        //1
animation.AddBitmap(NejiCombo26, RGB(0, 128, 128));        //2
animation.AddBitmap(NejiCombo26, RGB(0, 128, 128));        //3
animation.AddBitmap(NejiCombo26, RGB(0, 128, 128));        //4
animation.AddBitmap(NejiCombo27, RGB(0, 128, 128));        //5
animation.AddBitmap(NejiCombo27, RGB(0, 128, 128));        //6
animation.AddBitmap(NejiCombo27, RGB(0, 128, 128));        //7
animation.AddBitmap(NejiCombo28, RGB(0, 128, 128));        //8
animation.AddBitmap(NejiCombo28, RGB(0, 128, 128));        //9
animation.AddBitmap(NejiCombo28, RGB(0, 128, 128));        //160
animation.AddBitmap(NejiCombo29, RGB(0, 128, 128));        //1
animation.AddBitmap(NejiCombo29, RGB(0, 128, 128));        //2
animation.AddBitmap(NejiCombo29, RGB(0, 128, 128));        //3
animation.AddBitmap(NejiCombo30, RGB(0, 128, 128));        //4
animation.AddBitmap(NejiCombo30, RGB(0, 128, 128));        //5
animation.AddBitmap(NejiCombo30, RGB(0, 128, 128));        //6
animation.AddBitmap(NejiCombo31, RGB(0, 128, 128));        //7
animation.AddBitmap(NejiCombo31, RGB(0, 128, 128));        //8
animation.AddBitmap(NejiCombo31, RGB(0, 128, 128));        //9
animation.AddBitmap(NejiCombo32, RGB(0, 128, 128));        //170
animation.AddBitmap(NejiCombo32, RGB(0, 128, 128));        //1
animation.AddBitmap(NejiCombo32, RGB(0, 128, 128));        //2
animation.AddBitmap(NejiCombo33, RGB(0, 128, 128));        //3
animation.AddBitmap(NejiCombo33, RGB(0, 128, 128));        //4
animation.AddBitmap(NejiCombo33, RGB(0, 128, 128));        //5
animation.AddBitmap(NejiCombo34, RGB(0, 128, 128));        //6
animation.AddBitmap(NejiCombo34, RGB(0, 128, 128));        //7
animation.AddBitmap(NejiCombo34, RGB(0, 128, 128));        //8
animation.AddBitmap(NejiCombo35, RGB(0, 128, 128));        //9
animation.AddBitmap(NejiCombo35, RGB(0, 128, 128));        //180
animation.AddBitmap(NejiCombo35, RGB(0, 128, 128));        //1
animation.AddBitmap(NejiHurt3, RGB(0, 128, 128));          //2
animation.AddBitmap(NejiHurt3, RGB(0, 128, 128));          //3
animation.AddBitmap(NejiHurt3, RGB(0, 128, 128));          //4
animation.AddBitmap(NejiHurt4, RGB(0, 128, 128));          //5
animation.AddBitmap(NejiHurt4, RGB(0, 128, 128));          //6
animation.AddBitmap(NejiHurt4, RGB(0, 128, 128));          //7
animation.AddBitmap(NejiHurt3, RGB(0, 128, 128));          //8
animation.AddBitmap(NejiHurt3, RGB(0, 128, 128));          //9
animation.AddBitmap(NejiHurt3, RGB(0, 128, 128));          //190
animation.AddBitmap(NejiHurt3, RGB(0, 128, 128));          //1
animation.AddBitmap(NejiHurt3, RGB(0, 128, 128));          //2
animation.AddBitmap(NejiHurt4, RGB(0, 128, 128));          //3
animation.AddBitmap(NejiHurt4, RGB(0, 128, 128));          //4
animation.AddBitmap(NejiHurt4, RGB(0, 128, 128));          //5
animation.AddBitmap(NejiHurt4, RGB(0, 128, 128));          //6
animation.AddBitmap(NejiDie5, RGB(0, 128, 128));           //7
animation.AddBitmap(NejiDie5, RGB(0, 128, 128));           //8
animation.AddBitmap(NejiDie5, RGB(0, 128, 128));           //9
animation.AddBitmap(NejiDie5, RGB(0, 128, 128));           //200
animation.AddBitmap(NejiDie5, RGB(0, 128, 128));           //1
animation.AddBitmap(NejiDie5, RGB(0, 128, 128));           //2
```

```
animation.AddBitmap(NejiDie6, RGB(0, 128, 128));                    //3
animation.AddBitmap(NejiDie6, RGB(0, 128, 128));                    //4
animation.AddBitmap(NejiDie6, RGB(0, 128, 128));                    //5
animation.AddBitmap(NejiDie7, RGB(0, 128, 128));                    //6
animation.AddBitmap(NejiDie7, RGB(0, 128, 128));                    //7
animation.AddBitmap(NejiDie7, RGB(0, 128, 128));                    //8
animation.AddBitmap(NejiDie8, RGB(0, 128, 128));                    //9
animation.AddBitmap(NejiDie8, RGB(0, 128, 128));                    //210
animation.AddBitmap(NejiDie8, RGB(0, 128, 128));                    //1
animation.AddBitmap(NejiDie8, RGB(0, 128, 128));                    //2
animation.AddBitmap(NejiDie8, RGB(0, 128, 128));                    //3
animation.AddBitmap(NejiDie8, RGB(0, 128, 128));                    //4
animation.AddBitmap(NejiDie8, RGB(0, 128, 128));                    //5
animation.AddBitmap(NejiDie8, RGB(0, 128, 128));                    //6
animation.AddBitmap(NejiDie8, RGB(0, 128, 128));                    //7
SkillMovie.AddBitmap(NejiSkill1, RGB(0, 128, 128));                 //0
SkillMovie.AddBitmap(NejiSkill1, RGB(0, 128, 128));                 //1
SkillMovie.AddBitmap(NejiSkill1, RGB(0, 128, 128));                 //2
SkillMovie.AddBitmap(NejiSkill2, RGB(0, 128, 128));                 //3
SkillMovie.AddBitmap(NejiSkill2, RGB(0, 128, 128));                 //4
SkillMovie.AddBitmap(NejiSkill2, RGB(0, 128, 128));                 //5
SkillMovie.AddBitmap(NejiSkill3, RGB(0, 128, 128));                 //6
SkillMovie.AddBitmap(NejiSkill3, RGB(0, 128, 128));                 //7
SkillMovie.AddBitmap(NejiSkill3, RGB(0, 128, 128));                 //8
SkillMovie.AddBitmap(NejiSkill4, RGB(0, 128, 128));                 //9
SkillMovie.AddBitmap(NejiSkill4, RGB(0, 128, 128));                 //10
SkillMovie.AddBitmap(NejiSkill4, RGB(0, 128, 128));                 //1
SkillMovie.AddBitmap(NejiSkill5, RGB(0, 128, 128));                 //2
SkillMovie.AddBitmap(NejiSkill5, RGB(0, 128, 128));                 //3
SkillMovie.AddBitmap(NejiSkill5, RGB(0, 128, 128));                 //4
SkillMovie.AddBitmap(NejiSkill6, RGB(0, 128, 128));                 //5
SkillMovie.AddBitmap(NejiSkill6, RGB(0, 128, 128));                 //6
SkillMovie.AddBitmap(NejiSkill6, RGB(0, 128, 128));                 //7
SkillMovie.AddBitmap(NejiSkill7, RGB(0, 128, 128));                 //8
SkillMovie.AddBitmap(NejiSkill7, RGB(0, 128, 128));                 //9
SkillMovie.AddBitmap(NejiSkill7, RGB(0, 128, 128));                 //20
SkillMovie.AddBitmap(NejiSkill8, RGB(0, 128, 128));                 //1
SkillMovie.AddBitmap(NejiSkill8, RGB(0, 128, 128));                 //2
SkillMovie.AddBitmap(NejiSkill8, RGB(0, 128, 128));                 //3
SkillMovie.AddBitmap(NejiSkill9, RGB(0, 128, 128));                 //4
SkillMovie.AddBitmap(NejiSkill9, RGB(0, 128, 128));                 //5
SkillMovie.AddBitmap(NejiSkill9, RGB(0, 128, 128));                 //6
SkillMovie.AddBitmap(NejiSkill10, RGB(0, 128, 128));                //7
SkillMovie.AddBitmap(NejiSkill10, RGB(0, 128, 128));                //8
SkillMovie.AddBitmap(NejiSkill10, RGB(0, 128, 128));                //9
SkillMovie.AddBitmap(NejiSkill11, RGB(0, 128, 128));                //30
SkillMovie.AddBitmap(NejiSkill11, RGB(0, 128, 128));                //1
SkillMovie.AddBitmap(NejiSkill11, RGB(0, 128, 128));                //2
SkillMovie.AddBitmap(NejiSkill12, RGB(0, 128, 128));                //3
SkillMovie.AddBitmap(NejiSkill12, RGB(0, 128, 128));                //4
SkillMovie.AddBitmap(NejiSkill12, RGB(0, 128, 128));                //5
SkillMovie.AddBitmap(NejiSkill13, RGB(0, 128, 128));                //6
SkillMovie.AddBitmap(NejiSkill13, RGB(0, 128, 128));                //7
SkillMovie.AddBitmap(NejiSkill13, RGB(0, 128, 128));                //8
SkillMovie.AddBitmap(NejiSkill33, RGB(0, 128, 128));                //9
SkillMovie.AddBitmap(NejiSkill33, RGB(0, 128, 128));                //40
SkillMovie.AddBitmap(NejiSkill33, RGB(0, 128, 128));                //1
SkillMovie.AddBitmap(NejiSkill34, RGB(0, 128, 128));                //2
SkillMovie.AddBitmap(NejiSkill34, RGB(0, 128, 128));                //3
SkillMovie.AddBitmap(NejiSkill34, RGB(0, 128, 128));                //4
SkillMovie.AddBitmap(NejiSkill35, RGB(0, 128, 128));                //5
SkillMovie.AddBitmap(NejiSkill35, RGB(0, 128, 128));                //6
SkillMovie.AddBitmap(NejiSkill35, RGB(0, 128, 128));                //7
SkillMovie.AddBitmap(NejiSkill36, RGB(0, 128, 128));                //8
SkillMovie.AddBitmap(NejiSkill36, RGB(0, 128, 128));          //9
SkillMovie.AddBitmap(NejiSkill36, RGB(0, 128, 128));          //50
SkillMovie.AddBitmap(NejiSkill37, RGB(0, 128, 128));          //1
SkillMovie.AddBitmap(NejiSkill37, RGB(0, 128, 128));          //2
SkillMovie.AddBitmap(NejiSkill37, RGB(0, 128, 128));          //3
SkillMovie.AddBitmap(NejiSkill38, RGB(0, 128, 128));          //4
SkillMovie.AddBitmap(NejiSkill38, RGB(0, 128, 128));          //5
SkillMovie.AddBitmap(NejiSkill38, RGB(0, 128, 128));          //6
SkillMovie.AddBitmap(NejiSkill39, RGB(0, 128, 128));          //7
SkillMovie.AddBitmap(NejiSkill39, RGB(0, 128, 128));          //8
SkillMovie.AddBitmap(NejiSkill39, RGB(0, 128, 128));          //9
SkillMovie.AddBitmap(NejiSkill40, RGB(0, 128, 128));          //60
```

```
SkillMovie.AddBitmap(NejiSkill40, RGB(0, 128, 128));          //1
SkillMovie.AddBitmap(NejiSkill40, RGB(0, 128, 128));          //2
SkillMovie.AddBitmap(NejiSkill41, RGB(0, 128, 128));          //3
SkillMovie.AddBitmap(NejiSkill41, RGB(0, 128, 128));          //4
SkillMovie.AddBitmap(NejiSkill41, RGB(0, 128, 128));          //5
SkillMovie.AddBitmap(NejiSkill42, RGB(0, 128, 128));          //6
SkillMovie.AddBitmap(NejiSkill42, RGB(0, 128, 128));          //7
SkillMovie.AddBitmap(NejiSkill42, RGB(0, 128, 128));          //8
SkillMovie.AddBitmap(NejiSkill43, RGB(0, 128, 128));          //9
SkillMovie.AddBitmap(NejiSkill43, RGB(0, 128, 128));          //70
SkillMovie.AddBitmap(NejiSkill43, RGB(0, 128, 128));          //1
SkillMovie.AddBitmap(NejiSkill44, RGB(0, 128, 128));          //2
SkillMovie.AddBitmap(NejiSkill44, RGB(0, 128, 128));          //3
SkillMovie.AddBitmap(NejiSkill44, RGB(0, 128, 128));          //4
SkillMovie.AddBitmap(NejiSkill45, RGB(0, 128, 128));          //5
SkillMovie.AddBitmap(NejiSkill45, RGB(0, 128, 128));          //6
SkillMovie.AddBitmap(NejiSkill45, RGB(0, 128, 128));          //7
SkillsuccessMovie.AddBitmap(NejiSkill14, RGB(0, 128, 128));   //0
SkillsuccessMovie.AddBitmap(NejiSkill14, RGB(0, 128, 128));   //1
SkillsuccessMovie.AddBitmap(NejiSkill14, RGB(0, 128, 128));   //2
SkillsuccessMovie.AddBitmap(NejiSkill15, RGB(0, 128, 128));   //3
SkillsuccessMovie.AddBitmap(NejiSkill15, RGB(0, 128, 128));   //4
SkillsuccessMovie.AddBitmap(NejiSkill15, RGB(0, 128, 128));   //5
SkillsuccessMovie.AddBitmap(NejiSkill16, RGB(0, 128, 128));   //6
SkillsuccessMovie.AddBitmap(NejiSkill16, RGB(0, 128, 128));   //7
SkillsuccessMovie.AddBitmap(NejiSkill16, RGB(0, 128, 128));   //8
SkillsuccessMovie.AddBitmap(NejiSkill17, RGB(0, 128, 128));   //9
SkillsuccessMovie.AddBitmap(NejiSkill17, RGB(0, 128, 128));   //10
SkillsuccessMovie.AddBitmap(NejiSkill17, RGB(0, 128, 128));   //1
SkillsuccessMovie.AddBitmap(NejiSkill18, RGB(0, 128, 128));   //2
SkillsuccessMovie.AddBitmap(NejiSkill18, RGB(0, 128, 128));   //3
SkillsuccessMovie.AddBitmap(NejiSkill18, RGB(0, 128, 128));   //4
SkillsuccessMovie.AddBitmap(NejiSkill19, RGB(0, 128, 128));   //5
SkillsuccessMovie.AddBitmap(NejiSkill19, RGB(0, 128, 128));   //6
SkillsuccessMovie.AddBitmap(NejiSkill19, RGB(0, 128, 128));   //7
SkillsuccessMovie.AddBitmap(NejiSkill20, RGB(0, 128, 128));   //8
SkillsuccessMovie.AddBitmap(NejiSkill20, RGB(0, 128, 128));   //9
SkillsuccessMovie.AddBitmap(NejiSkill20, RGB(0, 128, 128));   //20
SkillsuccessMovie.AddBitmap(NejiSkill21, RGB(0, 128, 128));   //1
SkillsuccessMovie.AddBitmap(NejiSkill21, RGB(0, 128, 128));   //2
SkillsuccessMovie.AddBitmap(NejiSkill21, RGB(0, 128, 128));   //3
SkillsuccessMovie.AddBitmap(NejiSkill22, RGB(0, 128, 128));   //4
SkillsuccessMovie.AddBitmap(NejiSkill22, RGB(0, 128, 128));   //5
SkillsuccessMovie.AddBitmap(NejiSkill22, RGB(0, 128, 128));   //6
SkillsuccessMovie.AddBitmap(NejiSkill23, RGB(0, 128, 128));   //7
SkillsuccessMovie.AddBitmap(NejiSkill23, RGB(0, 128, 128));   //8
SkillsuccessMovie.AddBitmap(NejiSkill23, RGB(0, 128, 128));   //9
SkillsuccessMovie.AddBitmap(NejiSkill24, RGB(0, 128, 128));   //30
SkillsuccessMovie.AddBitmap(NejiSkill24, RGB(0, 128, 128));   //1
SkillsuccessMovie.AddBitmap(NejiSkill24, RGB(0, 128, 128));   //2
SkillsuccessMovie.AddBitmap(NejiSkill25, RGB(0, 128, 128));   //3
SkillsuccessMovie.AddBitmap(NejiSkill25, RGB(0, 128, 128));   //4
SkillsuccessMovie.AddBitmap(NejiSkill25, RGB(0, 128, 128));   //5
SkillsuccessMovie.AddBitmap(NejiSkill26, RGB(0, 128, 128));   //6
SkillsuccessMovie.AddBitmap(NejiSkill26, RGB(0, 128, 128));   //7
SkillsuccessMovie.AddBitmap(NejiSkill26, RGB(0, 128, 128));   //8
SkillsuccessMovie.AddBitmap(NejiSkill27, RGB(0, 128, 128));   //9
SkillsuccessMovie.AddBitmap(NejiSkill27, RGB(0, 128, 128));   //40
SkillsuccessMovie.AddBitmap(NejiSkill27, RGB(0, 128, 128));   //1
SkillsuccessMovie.AddBitmap(NejiSkill28, RGB(0, 128, 128));   //2
SkillsuccessMovie.AddBitmap(NejiSkill28, RGB(0, 128, 128));   //3
SkillsuccessMovie.AddBitmap(NejiSkill28, RGB(0, 128, 128));   //4
SkillsuccessMovie.AddBitmap(NejiSkill29, RGB(0, 128, 128));   //5
SkillsuccessMovie.AddBitmap(NejiSkill29, RGB(0, 128, 128));   //6
SkillsuccessMovie.AddBitmap(NejiSkill29, RGB(0, 128, 128));   //7
SkillsuccessMovie.AddBitmap(NejiSkill30, RGB(0, 128, 128));   //8
SkillsuccessMovie.AddBitmap(NejiSkill30, RGB(0, 128, 128));   //9
SkillsuccessMovie.AddBitmap(NejiSkill30, RGB(0, 128, 128));   //50
SkillsuccessMovie.AddBitmap(NejiSkill31, RGB(0, 128, 128));   //1
SkillsuccessMovie.AddBitmap(NejiSkill31, RGB(0, 128, 128));   //2
SkillsuccessMovie.AddBitmap(NejiSkill31, RGB(0, 128, 128));   //3
SkillsuccessMovie.AddBitmap(NejiSkill32, RGB(0, 128, 128));   //4
SkillsuccessMovie.AddBitmap(NejiSkill32, RGB(0, 128, 128));   //5
SkillsuccessMovie.AddBitmap(NejiSkill32, RGB(0, 128, 128));   //6
SkillsuccessMovie.AddBitmap(NejiSkill46, RGB(0, 128, 128));   //7
SkillsuccessMovie.AddBitmap(NejiSkill46, RGB(0, 128, 128));   //8
```

```
		SkillsuccessMovie.AddBitmap(NejiSkill46, RGB(0, 128, 128));	//9
		SkillsuccessMovie.AddBitmap(NejiSkill47, RGB(0, 128, 128));	//60
		SkillsuccessMovie.AddBitmap(NejiSkill47, RGB(0, 128, 128));	//1
		SkillsuccessMovie.AddBitmap(NejiSkill47, RGB(0, 128, 128));	//2
		SkillsuccessMovie.AddBitmap(NejiSkill48, RGB(0, 128, 128));	//3
		SkillsuccessMovie.AddBitmap(NejiSkill48, RGB(0, 128, 128));	//4
		SkillsuccessMovie.AddBitmap(NejiSkill48, RGB(0, 128, 128));	//5
		SkillsuccessMovie.AddBitmap(NejiSkill49, RGB(0, 128, 128));	//6
		SkillsuccessMovie.AddBitmap(NejiSkill49, RGB(0, 128, 128));	//7
		SkillsuccessMovie.AddBitmap(NejiSkill49, RGB(0, 128, 128));	//8
		SkillsuccessMovie.AddBitmap(NejiSkill50, RGB(0, 128, 128));	//9
		SkillsuccessMovie.AddBitmap(NejiSkill50, RGB(0, 128, 128));	//70
		SkillsuccessMovie.AddBitmap(NejiSkill50, RGB(0, 128, 128));	//1
		SkillsuccessMovie.AddBitmap(NejiSkill51, RGB(0, 128, 128));	//2
		SkillsuccessMovie.AddBitmap(NejiSkill51, RGB(0, 128, 128));	//3
		SkillsuccessMovie.AddBitmap(NejiSkill51, RGB(0, 128, 128));	//4
		SkillsuccessMovie.AddBitmap(NejiSkill52, RGB(0, 128, 128));	//5
		SkillsuccessMovie.AddBitmap(NejiSkill52, RGB(0, 128, 128));	//6
		SkillsuccessMovie.AddBitmap(NejiSkill52, RGB(0, 128, 128));	//7
		SkillsuccessMovie.AddBitmap(NejiSkill53, RGB(0, 128, 128));	//8
		SkillsuccessMovie.AddBitmap(NejiSkill53, RGB(0, 128, 128));	//9
		SkillsuccessMovie.AddBitmap(NejiSkill53, RGB(0, 128, 128));	//80
		SkillsuccessMovie.AddBitmap(NejiSkill54, RGB(0, 128, 128));	//1
		SkillsuccessMovie.AddBitmap(NejiSkill54, RGB(0, 128, 128));	//2
		SkillsuccessMovie.AddBitmap(NejiSkill54, RGB(0, 128, 128));	//3
		SkillsuccessMovie.AddBitmap(NejiSkill55, RGB(0, 128, 128));	//4
		SkillsuccessMovie.AddBitmap(NejiSkill55, RGB(0, 128, 128));	//5
		SkillsuccessMovie.AddBitmap(NejiSkill55, RGB(0, 128, 128));	//6
		SkillsuccessMovie.AddBitmap(NejiSkill56, RGB(0, 128, 128));	//7
		SkillsuccessMovie.AddBitmap(NejiSkill56, RGB(0, 128, 128));	//8
		SkillsuccessMovie.AddBitmap(NejiSkill56, RGB(0, 128, 128));	//9
		SkillsuccessMovie.AddBitmap(NejiSkill57, RGB(0, 128, 128));	//90
		SkillsuccessMovie.AddBitmap(NejiSkill57, RGB(0, 128, 128));	//1
		SkillsuccessMovie.AddBitmap(NejiSkill57, RGB(0, 128, 128));	//2
		SkillsuccessMovie.AddBitmap(NejiSkill58, RGB(0, 128, 128));	//3
		SkillsuccessMovie.AddBitmap(NejiSkill58, RGB(0, 128, 128));	//4
		SkillsuccessMovie.AddBitmap(NejiSkill58, RGB(0, 128, 128));	//5
		SkillsuccessMovie.AddBitmap(NejiSkill59, RGB(0, 128, 128));	//6
		SkillsuccessMovie.AddBitmap(NejiSkill59, RGB(0, 128, 128));	//7
		SkillsuccessMovie.AddBitmap(NejiSkill59, RGB(0, 128, 128));	//8
		SkillsuccessMovie.AddBitmap(NejiSkill60, RGB(0, 128, 128));	//9
		SkillsuccessMovie.AddBitmap(NejiSkill60, RGB(0, 128, 128));	//100
		SkillsuccessMovie.AddBitmap(NejiSkill60, RGB(0, 128, 128));	//1
		SkillsuccessMovie.AddBitmap(NejiSkill61, RGB(0, 128, 128));	//2
		SkillsuccessMovie.AddBitmap(NejiSkill61, RGB(0, 128, 128));	//3
		SkillsuccessMovie.AddBitmap(NejiSkill61, RGB(0, 128, 128));	//4
		SkillsuccessMovie.AddBitmap(NejiSkill62, RGB(0, 128, 128));	//5
		SkillsuccessMovie.AddBitmap(NejiSkill62, RGB(0, 128, 128));	//6
		SkillsuccessMovie.AddBitmap(NejiSkill62, RGB(0, 128, 128));	//7
		SkillsuccessMovie.AddBitmap(NejiSkill63, RGB(0, 128, 128));	//8
		SkillsuccessMovie.AddBitmap(NejiSkill63, RGB(0, 128, 128));	//9
		SkillsuccessMovie.AddBitmap(NejiSkill63, RGB(0, 128, 128));	//0
		SkillsuccessMovie.AddBitmap(NejiSkill64, RGB(0, 128, 128));	//1
		SkillsuccessMovie.AddBitmap(NejiSkill64, RGB(0, 128, 128));	//2
		SkillsuccessMovie.AddBitmap(NejiSkill64, RGB(0, 128, 128));	//3
		Movie.LoadBitmap(NejiSkillUI, RGB(0, 128, 128));
		SkillMovie.SetDelayCount(1);
		SkillMovie.Reset();
		SkillsuccessMovie.SetDelayCount(1);
		SkillsuccessMovie.Reset();
	}
}
```

Shikamaru.h

```
#ifndef SHIKAMARU_H
#define SHIKAMARU_H
#include "EnemyAI.h"
namespace game_framework {
	class Shikamaru : public EnemyAI
	{
	public:
		Shikamaru();
		~Shikamaru();
		int	GetX1();
		int	GetY1();
		int	GetX2();
		int	GetY2();
```

```
		void AttackMovie();
		void RunSkillsuccessMovie();
		void Initialize();				// 設定角色為初始值
		void LoadBitmap();				// 載入圖形
		void UI(bool Hit);
		void SetpicXY();
		void SetSasukeDistance();
		void OnShow();
	private:
		int picx, picy;
		int Skilldelay;
		int SasukeDistance;
		int _direction, _distance;
	};
}
#endif
```

Shikamaru.cpp

```
#include "stdafx.h"
#include "Resource.h"
#include <mmsystem.h>
#include <ddraw.h>
#include "audio.h"
#include "gamelib.h"
#include "Shikamaru.h"

namespace game_framework {
	Shikamaru::Shikamaru()
	{
		Initialize();
	}
	Shikamaru::~Shikamaru()
	{
	}
	void Shikamaru::AttackMovie()
	{
		//////////////////////////////////////////////////////////播放技能動畫//////////////////////////////////////////////
		if (Movie.isStart())
		{
			Movie.OnMove();
			Player->SetMove(false);
		}
		//////////////////////////////////////////////////////////技能判定//////////////////////////////////////////////////
		if (!Movie.isStart())
		{
			if (!Skillsuccess) { SkillMovie.OnMove(); Player->SetMove(true); }
			else RunSkillsuccessMovie();
			if (SkillMovie.GetCurrentBitmapNumber() == 26 + direction * 27)
			{
				Move = true;
				isSkill = false;
				Movieflag = false;
				isDie = false;
				isAttacked = false;
				Skill = rand() % 10;//重新選擇攻擊方式
			}
		}
		//////////////////////////////////////////////////////////////////////////////////////////////////////////////////
	}
	void Shikamaru::RunSkillsuccessMovie()
	{
		//////////////////////////////////////////////////////////技能成功//////////////////////////////////////////////////
		if (SkillsuccessMovie.GetCurrentBitmapNumber() == 0+direction*43 && picy < 460)
		{
			picy += 20;//井野從空中落下
		}
		//////////////////////////////////////////////////////////井野攻擊//////////////////////////////////////////////////
		else if (SkillsuccessMovie.GetCurrentBitmapNumber() == 20 + direction * 43 || SkillsuccessMovie.GetCurrentBitmapNumber() == 29 +
		direction * 43)
		{
			shoot.push_back(new Shurikan(picx, picy- SkillsuccessMovie.Height(), direction));
			SkillsuccessMovie.OnMove();
		}
		//////////////////////////////////////////////////////////丁次落下//////////////////////////////////////////////////
		else if (SkillsuccessMovie.GetCurrentBitmapNumber() == 38 + direction * 43 && picy < 600)
		{
			picy += 20;
```

```
	}
	else if (SkillsuccessMovie.GetCurrentBitmapNumber() == 38 + direction * 43 && picy >= 600)
	{
		SetpicXY();
		SkillsuccessMovie.OnMove();
	}
	else if (SkillsuccessMovie.GetCurrentBitmapNumber() > 38 + direction * 43)
	{
		if (SkillsuccessMovie.GetCurrentBitmapNumber() == 42 + direction * 43) SkillsuccessMovie.Movetonum(39+ direction * 43);
		else SkillsuccessMovie.OnMove();
		if (picy < 460 && Skilldelay>=0)
		{
			picy += 20;
			picx -= SasukeDistance;
		}
		if (picy == 460) Player->SetAttacked(true);
		if (Skilldelay-- <= 0)
		{
			picy -= 20;//丁次離開
			picx -= SasukeDistance*2;
		}
		////////////////////////////////////技能結束//////////////////////////////
		if (Skilldelay <= 0 && picy<0)
		{
			Skillsuccess = false;
			isSkill = false;
			Move = true;
			Movieflag = false;
			isDie = false;
			isAttacked = false;
			Skilldelay = 33;
			SkillsuccessMovie.Reset();
			Player->SetDie();
			Skill = rand() % 10;
		}
	}
	else SkillsuccessMovie.OnMove();
	SkillsuccessMovie.SetTopLeft(picx, picy - SkillsuccessMovie.Height()*2);
}
int Shikamaru::GetX1()
{
	/////////////////////////////////////////////////////圖片 x 座標修正//////////////////////////////////////////
	if (isSkill && direction == 0) return x + (StandWidth - SkillMovie.Width()) * 2;
	if (direction == 1) return x;
	return x + (StandWidth - animation.Width()) * 2;
	////////////////////////////////////////////////////////////////////////////////////////////////////////////
}
int Shikamaru::GetY1()
{
	/////////////////////////////////////////////////////圖片 y 座標修正//////////////////////////////////////////
	if (isSkill) return y + (StandHeight - SkillMovie.Height()) * 2 + 30;
	return y + (StandHeight - animation.Height()) * 2;
	////////////////////////////////////////////////////////////////////////////////////////////////////////////
}
int Shikamaru::GetX2()
{
	if (isSkill) return GetX1() + SkillMovie.Width() * 2;
	return GetX1() + animation.Width() * 2;
}
int Shikamaru::GetY2()
{
	if (isSkill) return GetY1() + SkillMovie.Height() * 2 - 30;
	return GetY1() + animation.Height() * 2;
}
void Shikamaru::Initialize()
{
	//////////////////////////////////////////////////初始化//////////////////////////////////////////////////////
	floor = 460;
	const int X_POS = 500;
	const int Y_POS = 460 - 39;
	const int INITIAL_VELOCITY = 18;
	initial_velocity = INITIAL_VELOCITY;
	velocity = initial_velocity;
	x = X_POS;
	y = Y_POS;
	isThrow=Movieflag=Skillsuccess=isSkill=Wall=isAttack=isDie = isShooting = isAttacked = rising = isMovingLeft = isMovingRight =
```

```
	isMovingUp= false;
	step_size = 10;
	Move =isShow= true;
	attacknum = direction = 0;
	StandHeight = 42;
	StandWidth = 21;
	Total = 89, StandStart = 0, StandNum = 26, RunStart = 27, RunNum = 5, JumpStart = 33, JumpNum = 4; AttackStart = 38; AttackNum = 8,
	ThrowStart = 47, ThrowNum = 8,HurtStart=56,HurtNum=2,DieStart=59,DieNum=29;
	distance = 200;
	Playerdirection = 1;
	Blood = 128;
	delay = 2;
	Skilldelay = 33;
	Skill = rand() % 10;
	_direction = _distance = 1;
	Diedelay = 30;
	////////////////////////////////////////////////////////////////////////////////////////////////////////////////
}
void Shikamaru::UI(bool Hit)
{
	//////////////////////////////////////////依照對方位置決定面向哪裡//////////////////////////////////////////////
	if (Player->GetX2() < GetX1())
	{
		if (!GetAttack() && !GetAttacked() && !GetDie() && !GetisSkill() && GetAttacknum() == 0)
		{
			_direction = 1;
			SetDirection(0);
		}
	}
	else if (Player->GetX1() > GetX2())
	{
		if (!GetAttack() && !GetAttacked() && !GetDie() && !GetisSkill() && GetAttacknum() == 0)
		{
			_direction = 0;
			SetDirection(1);
		}
	}
	if (_direction == 0) _distance = Player->GetX2() - GetX1();
	else _distance = Player->GetX1() - GetX2();
	_distance = abs(_distance);
	////////////////////////////////////////////////判斷要移動的距離//////////////////////////////////////////////////
	if (!isAttack) distance = 200 - _distance;
	if (!isAttack) Playerdirection = _direction;
	if (!Player->GetSkillsuccess())
	{
		if (isAttacked)
		{
			isSkill = false;
			Move = true;
			Movieflag = false;
		}
		//////////////////////////////////////////////////////技能施放//////////////////////////////////////////////////
		else if (Skill == 9 && times <= 0 && Move && distance < 190 || isSkill)
		{
			if (!Movieflag)
			{
				if (Player->GetX1() < GetX2()) direction = 0;
				else if (Player->GetX2() > GetX1()) direction = 1;
				SkillMovie.Movetonum(0 + direction * 27);
				Movieflag = true;
				Movie.Init();
				CAudio::Instance()->Play(31);
			}
			isMovingRight = false;
			isMovingLeft = false;
			isSkill = true;
			Move = false;
			AttackMovie();
			times = rand() % 75 + 30;
		}
		//////////////////////////////////////////////////////攻擊及移動判定//////////////////////////////////////////////
		else if (times <= 0 && Move)
		{
			if (distance < 200 || Wall)
			{
				////////////////////////////////////判斷是否走到牆壁//////////////////////////////////////////
```

```
                                if (Playerdirection == 1 && GetX2() + distance >= 640&&!Wall)
                                {
                                        Playerdirection = 0;
                                        distance += 200 - (GetX2() + distance - 640);//移動到另一邊
                                        Wall = true;
                                }
                                else if (Playerdirection == 0 && GetX1() - distance <= 0&& !Wall)
                                {
                                        Playerdirection = 1;
                                        distance += 200 + (GetX1() - distance);
                                        Wall = true;
                                }
                                ////////////////////////////////與佐助拉開 200 的距離//////////////////////////////////////////////
                                if (Playerdirection == 1)
                                {
                                        isMovingRight = true;
                                        isMovingLeft = false;
                                        distance -= 7;
                                }
                                else
                                {
                                        isMovingLeft = true;
                                        isMovingRight = false;
                                        distance -= 7;
                                }
                                ////////////////////////////////////////////////////////////////////////////////////////////////////
                        }
                        isAttack = true;
                        ////////////////////////////////////////攻擊//////////////////////////////////////////////////////
                        if (distance <= 0)
                        {
                                isMovingRight = false;
                                isMovingLeft = false;
                                if (_direction == 1) direction = 0;
                                else direction = 1;
                                isThrow = true;
                                times = rand() % 75 + 30;
                                isAttack = false;
                                Wall = false;
                        }
                        ////////////////////////////////////////////////////////////////////////////////////
                }
                ////////////////////////////////////////////////技能擊中判定//////////////////////////////////////////
                if (!Player->GetisMolding() && !Player->GetisSkill() && GetisSkill() && Hit && GetSkillMovie().GetCurrentBitmapNumber() > 7 +
                GetDirection() * 27 && GetSkillMovie().GetCurrentBitmapNumber() < 17 + GetDirection() * 27 && !GetSkillsuccess())
                {
                        SetpicXY();
                        SetSasukeX((Player->GetX1() + Player->GetX2()) / 2);
                        Player->SetMove(false);
                        SetSkillsuccess();
                        SetSasukeDistance();
                        GetSkillsuccessMovie()->Movetonum(0 + GetDirection() * 43);
                }
                ////////////////////////////////////////////////////////////////////////////////////////////////////////////
                if (Move) times--;
        }
}
void Shikamaru::SetpicXY()
{
        if (direction == 0) picx = GetX2() + 20;
        if (direction == 1) picx = GetX1() - 20;
        picy = 0;
}
void Shikamaru::SetSasukeDistance()
{
        SasukeDistance = (picx + SkillMovie.Width() - SasukeX)/23;
}
void Shikamaru::OnShow()
{
        ////////////////////////////////////////////顯示//////////////////////////////////////////////////////
        for (int count = 0; count < (int)shoot.size(); count++) shoot[count]->OnShow();
        if (!isSkill)
        {
                animation.SetTopLeft(GetX1(), GetY1());
                animation.OnShow(2);
        }
```

```
	else
	{
		SkillMovie.SetTopLeft(GetX1(), GetY1() - 30);
		SkillMovie.OnShow(2);
	}
	if (Skillsuccess) SkillsuccessMovie.OnShow(2);
}
void Shikamaru::LoadBitmap()
{
	/////////////////////////////////////////////////////////////讀取圖片/////////////////////////////////////////
	animation.AddBitmap(Shikamarustand1, RGB(255, 100, 100));		//0
	animation.AddBitmap(Shikamarustand1, RGB(255, 100, 100));		//1
	animation.AddBitmap(Shikamarustand1, RGB(255, 100, 100));		//2
	animation.AddBitmap(Shikamarustand1, RGB(255, 100, 100));		//3
	animation.AddBitmap(Shikamarustand1, RGB(255, 100, 100));		//4
	animation.AddBitmap(Shikamarustand1, RGB(255, 100, 100));		//5
	animation.AddBitmap(Shikamarustand2, RGB(255, 100, 100));		//6
	animation.AddBitmap(Shikamarustand2, RGB(255, 100, 100));		//7
	animation.AddBitmap(Shikamarustand2, RGB(255, 100, 100));		//8
	animation.AddBitmap(Shikamarustand2, RGB(255, 100, 100));		//9
	animation.AddBitmap(Shikamarustand2, RGB(255, 100, 100));		//10
	animation.AddBitmap(Shikamarustand2, RGB(255, 100, 100));		//1
	animation.AddBitmap(Shikamarustand3, RGB(255, 100, 100));		//2
	animation.AddBitmap(Shikamarustand3, RGB(255, 100, 100));		//3
	animation.AddBitmap(Shikamarustand3, RGB(255, 100, 100));		//4
	animation.AddBitmap(Shikamarustand3, RGB(255, 100, 100));		//5
	animation.AddBitmap(Shikamarustand3, RGB(255, 100, 100));		//6
	animation.AddBitmap(Shikamarustand3, RGB(255, 100, 100));		//7
	animation.AddBitmap(Shikamarustand4, RGB(255, 100, 100));		//8
	animation.AddBitmap(Shikamarustand4, RGB(255, 100, 100));		//9
	animation.AddBitmap(Shikamarustand4, RGB(255, 100, 100));		//20
	animation.AddBitmap(Shikamarustand4, RGB(255, 100, 100));		//1
	animation.AddBitmap(Shikamarustand4, RGB(255, 100, 100));		//2
	animation.AddBitmap(Shikamarustand4, RGB(255, 100, 100));		//3
	animation.AddBitmap(Shikamarustand4, RGB(255, 100, 100));		//4
	animation.AddBitmap(Shikamarustand4, RGB(255, 100, 100));		//5
	animation.AddBitmap(Shikamarustand4, RGB(255, 100, 100));		//6
	animation.AddBitmap(ShikamaruRun1, RGB(255, 100, 100));		//7
	animation.AddBitmap(ShikamaruRun1, RGB(255, 100, 100));		//8
	animation.AddBitmap(ShikamaruRun1, RGB(255, 100, 100));		//9
	animation.AddBitmap(ShikamaruRun2, RGB(255, 100, 100));		//30
	animation.AddBitmap(ShikamaruRun2, RGB(255, 100, 100));		//1
	animation.AddBitmap(ShikamaruRun2, RGB(255, 100, 100));		//2
	animation.AddBitmap(ShikamaruJump1, RGB(255, 100, 100));		//3
	animation.AddBitmap(ShikamaruJump2, RGB(255, 100, 100));		//4
	animation.AddBitmap(ShikamaruJump1, RGB(255, 100, 100));		//5
	animation.AddBitmap(ShikamaruJump1, RGB(255, 100, 100));		//6
	animation.AddBitmap(ShikamaruJump1, RGB(255, 100, 100));		//7
	animation.AddBitmap(ShikamarujumpKick1, RGB(255, 100, 100));	//8
	animation.AddBitmap(ShikamarujumpKick1, RGB(255, 100, 100));	//9
	animation.AddBitmap(ShikamarujumpKick1, RGB(255, 100, 100));	//40
	animation.AddBitmap(ShikamarujumpKick2, RGB(255, 100, 100));	//1
	animation.AddBitmap(ShikamarujumpKick2, RGB(255, 100, 100));	//2
	animation.AddBitmap(ShikamarujumpKick2, RGB(255, 100, 100));	//3
	animation.AddBitmap(ShikamarujumpKick3, RGB(255, 100, 100));	//4
	animation.AddBitmap(ShikamarujumpKick3, RGB(255, 100, 100));	//5
	animation.AddBitmap(ShikamarujumpKick3, RGB(255, 100, 100));	//6
	animation.AddBitmap(Shikamaruthrow1, RGB(255, 100, 100));		//7
	animation.AddBitmap(Shikamaruthrow1, RGB(255, 100, 100));		//8
	animation.AddBitmap(Shikamaruthrow1, RGB(255, 100, 100));		//9
	animation.AddBitmap(Shikamaruthrow2, RGB(255, 100, 100));		//50
	animation.AddBitmap(Shikamaruthrow2, RGB(255, 100, 100));		//1
	animation.AddBitmap(Shikamaruthrow2, RGB(255, 100, 100));		//2
	animation.AddBitmap(Shikamaruthrow3, RGB(255, 100, 100));		//3
	animation.AddBitmap(Shikamaruthrow3, RGB(255, 100, 100));		//4
	animation.AddBitmap(Shikamaruthrow3, RGB(255, 100, 100));		//5
	animation.AddBitmap(ShikamaruDie1, RGB(255, 100, 100));		//6
	animation.AddBitmap(ShikamaruDie1, RGB(255, 100, 100));		//7
	animation.AddBitmap(ShikamaruDie1, RGB(255, 100, 100));		//8
	animation.AddBitmap(ShikamaruDie1, RGB(255, 100, 100));		//9
	animation.AddBitmap(ShikamaruDie1, RGB(255, 100, 100));		//60
	animation.AddBitmap(ShikamaruDie1, RGB(255, 100, 100));		//1
	animation.AddBitmap(ShikamaruDie1, RGB(255, 100, 100));		//2
	animation.AddBitmap(ShikamaruDie1, RGB(255, 100, 100));		//3
	animation.AddBitmap(ShikamaruDie1, RGB(255, 100, 100));		//4
	animation.AddBitmap(ShikamaruDie1, RGB(255, 100, 100));		//5
```

```
animation.AddBitmap(ShikamaruDie1, RGB(255, 100, 100));          //6
animation.AddBitmap(ShikamaruDie1, RGB(255, 100, 100));          //7
animation.AddBitmap(ShikamaruDie2, RGB(255, 100, 100));          //8
animation.AddBitmap(ShikamaruDie2, RGB(255, 100, 100));          //9
animation.AddBitmap(ShikamaruDie2, RGB(255, 100, 100));          //70
animation.AddBitmap(ShikamaruDie2, RGB(255, 100, 100));          //1
animation.AddBitmap(ShikamaruDie2, RGB(255, 100, 100));          //2
animation.AddBitmap(ShikamaruDie2, RGB(255, 100, 100));          //3
animation.AddBitmap(ShikamaruDie2, RGB(255, 100, 100));          //4
animation.AddBitmap(ShikamaruDie2, RGB(255, 100, 100));          //5
animation.AddBitmap(ShikamaruDie2, RGB(255, 100, 100));          //6
animation.AddBitmap(ShikamaruDie2, RGB(255, 100, 100));          //7
animation.AddBitmap(ShikamaruDie2, RGB(255, 100, 100));          //8
animation.AddBitmap(ShikamaruDie2, RGB(255, 100, 100));          //9
animation.AddBitmap(ShikamaruDie3, RGB(255, 100, 100));          //80
animation.AddBitmap(ShikamaruDie3, RGB(255, 100, 100));          //1
animation.AddBitmap(ShikamaruDie4, RGB(255, 100, 100));          //2
animation.AddBitmap(ShikamaruDie4, RGB(255, 100, 100));          //3
animation.AddBitmap(ShikamaruDie5, RGB(255, 100, 100));          //4
animation.AddBitmap(ShikamaruDie5, RGB(255, 100, 100));          //5
animation.AddBitmap(ShikamaruDie6, RGB(255, 100, 100));          //6
animation.AddBitmap(ShikamaruDie6, RGB(255, 100, 100));          //7
animation.AddBitmap(ShikamaruDie7, RGB(255, 100, 100));          //8
animation.AddBitmap(Shikamarustand5, RGB(255, 100, 100));        //89
animation.AddBitmap(Shikamarustand5, RGB(255, 100, 100));        //90
animation.AddBitmap(Shikamarustand5, RGB(255, 100, 100));        //1
animation.AddBitmap(Shikamarustand5, RGB(255, 100, 100));        //2
animation.AddBitmap(Shikamarustand5, RGB(255, 100, 100));        //3
animation.AddBitmap(Shikamarustand5, RGB(255, 100, 100));        //4
animation.AddBitmap(Shikamarustand6, RGB(255, 100, 100));        //5
animation.AddBitmap(Shikamarustand6, RGB(255, 100, 100));        //6
animation.AddBitmap(Shikamarustand6, RGB(255, 100, 100));        //7
animation.AddBitmap(Shikamarustand6, RGB(255, 100, 100));        //8
animation.AddBitmap(Shikamarustand6, RGB(255, 100, 100));        //9
animation.AddBitmap(Shikamarustand6, RGB(255, 100, 100));        //100
animation.AddBitmap(Shikamarustand7, RGB(255, 100, 100));        //1
animation.AddBitmap(Shikamarustand7, RGB(255, 100, 100));        //2
animation.AddBitmap(Shikamarustand7, RGB(255, 100, 100));        //3
animation.AddBitmap(Shikamarustand7, RGB(255, 100, 100));        //4
animation.AddBitmap(Shikamarustand7, RGB(255, 100, 100));        //5
animation.AddBitmap(Shikamarustand7, RGB(255, 100, 100));        //6
animation.AddBitmap(Shikamarustand8, RGB(255, 100, 100));        //7
animation.AddBitmap(Shikamarustand8, RGB(255, 100, 100));        //8
animation.AddBitmap(Shikamarustand8, RGB(255, 100, 100));        //9
animation.AddBitmap(Shikamarustand8, RGB(255, 100, 100));        //110
animation.AddBitmap(Shikamarustand8, RGB(255, 100, 100));        //1
animation.AddBitmap(Shikamarustand8, RGB(255, 100, 100));        //2
animation.AddBitmap(Shikamarustand8, RGB(255, 100, 100));        //3
animation.AddBitmap(Shikamarustand8, RGB(255, 100, 100));        //4
animation.AddBitmap(Shikamarustand8, RGB(255, 100, 100));        //5
animation.AddBitmap(ShikamaruRun3, RGB(255, 100, 100));          //6
animation.AddBitmap(ShikamaruRun3, RGB(255, 100, 100));          //7
animation.AddBitmap(ShikamaruRun3, RGB(255, 100, 100));          //8
animation.AddBitmap(ShikamaruRun4, RGB(255, 100, 100));          //9
animation.AddBitmap(ShikamaruRun4, RGB(255, 100, 100));          //120
animation.AddBitmap(ShikamaruRun4, RGB(255, 100, 100));          //1
animation.AddBitmap(ShikamaruJump3, RGB(255, 100, 100));         //2
animation.AddBitmap(ShikamaruJump4, RGB(255, 100, 100));         //3
animation.AddBitmap(ShikamaruJump3, RGB(255, 100, 100));         //4
animation.AddBitmap(ShikamaruJump3, RGB(255, 100, 100));         //5
animation.AddBitmap(ShikamaruJump3, RGB(255, 100, 100));         //6
animation.AddBitmap(ShikamarujumpKick4, RGB(255, 100, 100));     //7
animation.AddBitmap(ShikamarujumpKick4, RGB(255, 100, 100));     //8
animation.AddBitmap(ShikamarujumpKick4, RGB(255, 100, 100));     //9
animation.AddBitmap(ShikamarujumpKick5, RGB(255, 100, 100));     //130
animation.AddBitmap(ShikamarujumpKick5, RGB(255, 100, 100));     //1
animation.AddBitmap(ShikamarujumpKick5, RGB(255, 100, 100));     //2
animation.AddBitmap(ShikamarujumpKick6, RGB(255, 100, 100));     //3
animation.AddBitmap(ShikamarujumpKick6, RGB(255, 100, 100));     //4
animation.AddBitmap(ShikamarujumpKick6, RGB(255, 100, 100));     //5
animation.AddBitmap(Shikamaruthrow4, RGB(255, 100, 100));        //6
animation.AddBitmap(Shikamaruthrow4, RGB(255, 100, 100));        //7
animation.AddBitmap(Shikamaruthrow4, RGB(255, 100, 100));        //8
animation.AddBitmap(Shikamaruthrow5, RGB(255, 100, 100));        //9
animation.AddBitmap(Shikamaruthrow5, RGB(255, 100, 100));        //140
animation.AddBitmap(Shikamaruthrow5, RGB(255, 100, 100));        //1
```

```
animation.AddBitmap(Shikamaruthrow6, RGB(255, 100, 100));          //2
animation.AddBitmap(Shikamaruthrow6, RGB(255, 100, 100));          //3
animation.AddBitmap(Shikamaruthrow6, RGB(255, 100, 100));          //4
animation.AddBitmap(ShikamaruDie8, RGB(255, 100, 100));            //5
animation.AddBitmap(ShikamaruDie8, RGB(255, 100, 100));            //6
animation.AddBitmap(ShikamaruDie8, RGB(255, 100, 100));            //7
animation.AddBitmap(ShikamaruDie8, RGB(255, 100, 100));            //8
animation.AddBitmap(ShikamaruDie8, RGB(255, 100, 100));            //9
animation.AddBitmap(ShikamaruDie8, RGB(255, 100, 100));            //150
animation.AddBitmap(ShikamaruDie8, RGB(255, 100, 100));            //1
animation.AddBitmap(ShikamaruDie8, RGB(255, 100, 100));            //2
animation.AddBitmap(ShikamaruDie8, RGB(255, 100, 100));            //3
animation.AddBitmap(ShikamaruDie8, RGB(255, 100, 100));            //4
animation.AddBitmap(ShikamaruDie8, RGB(255, 100, 100));            //5
animation.AddBitmap(ShikamaruDie8, RGB(255, 100, 100));            //6
animation.AddBitmap(ShikamaruDie9, RGB(255, 100, 100));            //7
animation.AddBitmap(ShikamaruDie9, RGB(255, 100, 100));            //8
animation.AddBitmap(ShikamaruDie9, RGB(255, 100, 100));            //9
animation.AddBitmap(ShikamaruDie9, RGB(255, 100, 100));            //160
animation.AddBitmap(ShikamaruDie9, RGB(255, 100, 100));            //1
animation.AddBitmap(ShikamaruDie9, RGB(255, 100, 100));            //2
animation.AddBitmap(ShikamaruDie9, RGB(255, 100, 100));            //3
animation.AddBitmap(ShikamaruDie9, RGB(255, 100, 100));            //4
animation.AddBitmap(ShikamaruDie9, RGB(255, 100, 100));            //5
animation.AddBitmap(ShikamaruDie9, RGB(255, 100, 100));            //6
animation.AddBitmap(ShikamaruDie9, RGB(255, 100, 100));            //7
animation.AddBitmap(ShikamaruDie9, RGB(255, 100, 100));            //8
animation.AddBitmap(ShikamaruDie10, RGB(255, 100, 100));           //9
animation.AddBitmap(ShikamaruDie10, RGB(255, 100, 100));           //170
animation.AddBitmap(ShikamaruDie11, RGB(255, 100, 100));           //1
animation.AddBitmap(ShikamaruDie11, RGB(255, 100, 100));           //2
animation.AddBitmap(ShikamaruDie12, RGB(255, 100, 100));           //3
animation.AddBitmap(ShikamaruDie12, RGB(255, 100, 100));           //4
animation.AddBitmap(ShikamaruDie13, RGB(255, 100, 100));           //5
animation.AddBitmap(ShikamaruDie13, RGB(255, 100, 100));           //6
animation.AddBitmap(ShikamaruDie14, RGB(255, 100, 100));           //7
SkillMovie.AddBitmap(ShikamaruSkill1, RGB(255, 100, 100));         //0
SkillMovie.AddBitmap(ShikamaruSkill1, RGB(255, 100, 100));         //1
SkillMovie.AddBitmap(ShikamaruSkill1, RGB(255, 100, 100));         //2
SkillMovie.AddBitmap(ShikamaruSkill2, RGB(255, 100, 100));         //3
SkillMovie.AddBitmap(ShikamaruSkill2, RGB(255, 100, 100));         //4
SkillMovie.AddBitmap(ShikamaruSkill2, RGB(255, 100, 100));         //5
SkillMovie.AddBitmap(ShikamaruSkill2, RGB(255, 100, 100));         //6
SkillMovie.AddBitmap(ShikamaruSkill2, RGB(255, 100, 100));         //7
SkillMovie.AddBitmap(ShikamaruSkill3, RGB(255, 100, 100));         //8
SkillMovie.AddBitmap(ShikamaruSkill3, RGB(255, 100, 100));         //9
SkillMovie.AddBitmap(ShikamaruSkill3, RGB(255, 100, 100));         //10
SkillMovie.AddBitmap(ShikamaruSkill3, RGB(255, 100, 100));         //1
SkillMovie.AddBitmap(ShikamaruSkill3, RGB(255, 100, 100));         //2
SkillMovie.AddBitmap(ShikamaruSkill3, RGB(255, 100, 100));         //3
SkillMovie.AddBitmap(ShikamaruSkill3, RGB(255, 100, 100));         //4
SkillMovie.AddBitmap(ShikamaruSkill3, RGB(255, 100, 100));         //5
SkillMovie.AddBitmap(ShikamaruSkill3, RGB(255, 100, 100));         //6
SkillMovie.AddBitmap(ShikamaruSkill3, RGB(255, 100, 100));         //7
SkillMovie.AddBitmap(ShikamaruSkill1, RGB(255, 100, 100));         //8
SkillMovie.AddBitmap(ShikamaruSkill1, RGB(255, 100, 100));         //9
SkillMovie.AddBitmap(ShikamaruSkill1, RGB(255, 100, 100));         //20
SkillMovie.AddBitmap(ShikamaruSkill1, RGB(255, 100, 100));         //1
SkillMovie.AddBitmap(ShikamaruSkill1, RGB(255, 100, 100));         //2
SkillMovie.AddBitmap(ShikamaruSkill1, RGB(255, 100, 100));         //3
SkillMovie.AddBitmap(ShikamaruSkill1, RGB(255, 100, 100));         //4
SkillMovie.AddBitmap(ShikamaruSkill1, RGB(255, 100, 100));         //5
SkillMovie.AddBitmap(ShikamaruSkill1, RGB(255, 100, 100));         //6
SkillMovie.AddBitmap(ShikamaruSkill20, RGB(255, 100, 100));        //7
SkillMovie.AddBitmap(ShikamaruSkill20, RGB(255, 100, 100));        //8
SkillMovie.AddBitmap(ShikamaruSkill20, RGB(255, 100, 100));        //9
SkillMovie.AddBitmap(ShikamaruSkill21, RGB(255, 100, 100));        //30
SkillMovie.AddBitmap(ShikamaruSkill21, RGB(255, 100, 100));        //1
SkillMovie.AddBitmap(ShikamaruSkill21, RGB(255, 100, 100));        //2
SkillMovie.AddBitmap(ShikamaruSkill21, RGB(255, 100, 100));        //3
SkillMovie.AddBitmap(ShikamaruSkill21, RGB(255, 100, 100));        //4
SkillMovie.AddBitmap(ShikamaruSkill22, RGB(255, 100, 100));        //5
SkillMovie.AddBitmap(ShikamaruSkill22, RGB(255, 100, 100));        //6
SkillMovie.AddBitmap(ShikamaruSkill22, RGB(255, 100, 100));        //7
SkillMovie.AddBitmap(ShikamaruSkill22, RGB(255, 100, 100));        //8
SkillMovie.AddBitmap(ShikamaruSkill22, RGB(255, 100, 100));        //9
```

```
SkillMovie.AddBitmap(ShikamaruSkill22, RGB(255, 100, 100));		//1
SkillMovie.AddBitmap(ShikamaruSkill22, RGB(255, 100, 100));		//2
SkillMovie.AddBitmap(ShikamaruSkill22, RGB(255, 100, 100));		//3
SkillMovie.AddBitmap(ShikamaruSkill22, RGB(255, 100, 100));		//4
SkillMovie.AddBitmap(ShikamaruSkill22, RGB(255, 100, 100));		//5
SkillMovie.AddBitmap(ShikamaruSkill20, RGB(255, 100, 100));		//6
SkillMovie.AddBitmap(ShikamaruSkill20, RGB(255, 100, 100));		//7
SkillMovie.AddBitmap(ShikamaruSkill20, RGB(255, 100, 100));		//8
SkillMovie.AddBitmap(ShikamaruSkill20, RGB(255, 100, 100));		//9
SkillMovie.AddBitmap(ShikamaruSkill20, RGB(255, 100, 100));		//1
SkillMovie.AddBitmap(ShikamaruSkill20, RGB(255, 100, 100));		//2
SkillMovie.AddBitmap(ShikamaruSkill20, RGB(255, 100, 100));		//3
SkillMovie.AddBitmap(ShikamaruSkill20, RGB(255, 100, 100));		//4
SkillMovie.AddBitmap(ShikamaruSkill20, RGB(255, 100, 100));		//5
SkillsuccessMovie.AddBitmap(ShikamaruSkill4, RGB(255, 100, 100));		//0
SkillsuccessMovie.AddBitmap(ShikamaruSkill5, RGB(255, 100, 100));		//1
SkillsuccessMovie.AddBitmap(ShikamaruSkill5, RGB(255, 100, 100));		//2
SkillsuccessMovie.AddBitmap(ShikamaruSkill5, RGB(255, 100, 100));		//3
SkillsuccessMovie.AddBitmap(ShikamaruSkill6, RGB(255, 100, 100));		//4
SkillsuccessMovie.AddBitmap(ShikamaruSkill6, RGB(255, 100, 100));		//5
SkillsuccessMovie.AddBitmap(ShikamaruSkill6, RGB(255, 100, 100));		//6
SkillsuccessMovie.AddBitmap(ShikamaruSkill7, RGB(255, 100, 100));		//7
SkillsuccessMovie.AddBitmap(ShikamaruSkill7, RGB(255, 100, 100));		//8
SkillsuccessMovie.AddBitmap(ShikamaruSkill7, RGB(255, 100, 100));		//9
SkillsuccessMovie.AddBitmap(ShikamaruSkill8, RGB(255, 100, 100));		//10
SkillsuccessMovie.AddBitmap(ShikamaruSkill8, RGB(255, 100, 100));		//1
SkillsuccessMovie.AddBitmap(ShikamaruSkill8, RGB(255, 100, 100));		//2
SkillsuccessMovie.AddBitmap(ShikamaruSkill9, RGB(255, 100, 100));		//3
SkillsuccessMovie.AddBitmap(ShikamaruSkill9, RGB(255, 100, 100));		//4
SkillsuccessMovie.AddBitmap(ShikamaruSkill9, RGB(255, 100, 100));		//5
SkillsuccessMovie.AddBitmap(ShikamaruSkill10, RGB(255, 100, 100));	//6
SkillsuccessMovie.AddBitmap(ShikamaruSkill10, RGB(255, 100, 100));	//7
SkillsuccessMovie.AddBitmap(ShikamaruSkill10, RGB(255, 100, 100));	//8
SkillsuccessMovie.AddBitmap(ShikamaruSkill11, RGB(255, 100, 100));	//9
SkillsuccessMovie.AddBitmap(ShikamaruSkill11, RGB(255, 100, 100));	//20
SkillsuccessMovie.AddBitmap(ShikamaruSkill11, RGB(255, 100, 100));	//1
SkillsuccessMovie.AddBitmap(ShikamaruSkill12, RGB(255, 100, 100));	//2
SkillsuccessMovie.AddBitmap(ShikamaruSkill12, RGB(255, 100, 100));	//3
SkillsuccessMovie.AddBitmap(ShikamaruSkill12, RGB(255, 100, 100));	//4
SkillsuccessMovie.AddBitmap(ShikamaruSkill13, RGB(255, 100, 100));	//5
SkillsuccessMovie.AddBitmap(ShikamaruSkill13, RGB(255, 100, 100));	//6
SkillsuccessMovie.AddBitmap(ShikamaruSkill13, RGB(255, 100, 100));	//7
SkillsuccessMovie.AddBitmap(ShikamaruSkill14, RGB(255, 100, 100));	//8
SkillsuccessMovie.AddBitmap(ShikamaruSkill14, RGB(255, 100, 100));	//9
SkillsuccessMovie.AddBitmap(ShikamaruSkill14, RGB(255, 100, 100));	//30
SkillsuccessMovie.AddBitmap(ShikamaruSkill15, RGB(255, 100, 100));	//1
SkillsuccessMovie.AddBitmap(ShikamaruSkill15, RGB(255, 100, 100));	//2
SkillsuccessMovie.AddBitmap(ShikamaruSkill15, RGB(255, 100, 100));	//3
SkillsuccessMovie.AddBitmap(ShikamaruSkill15, RGB(255, 100, 100));	//4
SkillsuccessMovie.AddBitmap(ShikamaruSkill15, RGB(255, 100, 100));	//5
SkillsuccessMovie.AddBitmap(ShikamaruSkill15, RGB(255, 100, 100));	//6
SkillsuccessMovie.AddBitmap(ShikamaruSkill15, RGB(255, 100, 100));	//7
SkillsuccessMovie.AddBitmap(ShikamaruSkill4, RGB(255, 100, 100));		//8
SkillsuccessMovie.AddBitmap(ShikamaruSkill16, RGB(255, 100, 100));	//9
SkillsuccessMovie.AddBitmap(ShikamaruSkill17, RGB(255, 100, 100));	//40
SkillsuccessMovie.AddBitmap(ShikamaruSkill18, RGB(255, 100, 100));	//1
SkillsuccessMovie.AddBitmap(ShikamaruSkill19, RGB(255, 100, 100));	//2
SkillsuccessMovie.AddBitmap(ShikamaruSkill23, RGB(255, 100, 100));	//0
SkillsuccessMovie.AddBitmap(ShikamaruSkill24, RGB(255, 100, 100));	//1
SkillsuccessMovie.AddBitmap(ShikamaruSkill24, RGB(255, 100, 100));	//2
SkillsuccessMovie.AddBitmap(ShikamaruSkill24, RGB(255, 100, 100));	//3
SkillsuccessMovie.AddBitmap(ShikamaruSkill25, RGB(255, 100, 100));	//4
SkillsuccessMovie.AddBitmap(ShikamaruSkill25, RGB(255, 100, 100));	//5
SkillsuccessMovie.AddBitmap(ShikamaruSkill25, RGB(255, 100, 100));	//6
SkillsuccessMovie.AddBitmap(ShikamaruSkill26, RGB(255, 100, 100));	//7
SkillsuccessMovie.AddBitmap(ShikamaruSkill26, RGB(255, 100, 100));	//8
SkillsuccessMovie.AddBitmap(ShikamaruSkill26, RGB(255, 100, 100));	//9
SkillsuccessMovie.AddBitmap(ShikamaruSkill27, RGB(255, 100, 100));	//10
SkillsuccessMovie.AddBitmap(ShikamaruSkill27, RGB(255, 100, 100));	//1
SkillsuccessMovie.AddBitmap(ShikamaruSkill27, RGB(255, 100, 100));	//2
SkillsuccessMovie.AddBitmap(ShikamaruSkill28, RGB(255, 100, 100));	//3
SkillsuccessMovie.AddBitmap(ShikamaruSkill28, RGB(255, 100, 100));	//4
SkillsuccessMovie.AddBitmap(ShikamaruSkill28, RGB(255, 100, 100));	//5
SkillsuccessMovie.AddBitmap(ShikamaruSkill29, RGB(255, 100, 100));	//6
SkillsuccessMovie.AddBitmap(ShikamaruSkill29, RGB(255, 100, 100));	//7
SkillsuccessMovie.AddBitmap(ShikamaruSkill29, RGB(255, 100, 100));	//8
```

```
                SkillsuccessMovie.AddBitmap(ShikamaruSkill30, RGB(255, 100, 100));    //9
                SkillsuccessMovie.AddBitmap(ShikamaruSkill30, RGB(255, 100, 100));    //20
                SkillsuccessMovie.AddBitmap(ShikamaruSkill30, RGB(255, 100, 100));    //1
                SkillsuccessMovie.AddBitmap(ShikamaruSkill31, RGB(255, 100, 100));    //2
                SkillsuccessMovie.AddBitmap(ShikamaruSkill31, RGB(255, 100, 100));    //3
                SkillsuccessMovie.AddBitmap(ShikamaruSkill31, RGB(255, 100, 100));    //4
                SkillsuccessMovie.AddBitmap(ShikamaruSkill32, RGB(255, 100, 100));    //5
                SkillsuccessMovie.AddBitmap(ShikamaruSkill32, RGB(255, 100, 100));    //6
                SkillsuccessMovie.AddBitmap(ShikamaruSkill32, RGB(255, 100, 100));    //7
                SkillsuccessMovie.AddBitmap(ShikamaruSkill33, RGB(255, 100, 100));    //8
                SkillsuccessMovie.AddBitmap(ShikamaruSkill33, RGB(255, 100, 100));    //9
                SkillsuccessMovie.AddBitmap(ShikamaruSkill33, RGB(255, 100, 100));    //30
                SkillsuccessMovie.AddBitmap(ShikamaruSkill34, RGB(255, 100, 100));    //1
                SkillsuccessMovie.AddBitmap(ShikamaruSkill34, RGB(255, 100, 100));    //2
                SkillsuccessMovie.AddBitmap(ShikamaruSkill34, RGB(255, 100, 100));    //3
                SkillsuccessMovie.AddBitmap(ShikamaruSkill34, RGB(255, 100, 100));    //4
                SkillsuccessMovie.AddBitmap(ShikamaruSkill34, RGB(255, 100, 100));    //5
                SkillsuccessMovie.AddBitmap(ShikamaruSkill34, RGB(255, 100, 100));    //6
                SkillsuccessMovie.AddBitmap(ShikamaruSkill34, RGB(255, 100, 100));    //7
                SkillsuccessMovie.AddBitmap(ShikamaruSkill23, RGB(255, 100, 100));    //8
                SkillsuccessMovie.AddBitmap(ShikamaruSkill16, RGB(255, 100, 100));    //9
                SkillsuccessMovie.AddBitmap(ShikamaruSkill17, RGB(255, 100, 100));    //40
                SkillsuccessMovie.AddBitmap(ShikamaruSkill18, RGB(255, 100, 100));    //1
                SkillsuccessMovie.AddBitmap(ShikamaruSkill19, RGB(255, 100, 100));    //2
                Movie.LoadBitmap(ShikamaruSkillUI, RGB(255, 100, 100));
                SkillsuccessMovie.SetDelayCount(1);
                SkillsuccessMovie.Reset();
                SkillMovie.SetDelayCount(1);
                SkillMovie.Reset();
        }
}
```

Orochimaru.h

```
#ifndef OROCHIMARU_H
#define OROCHIMARU_H
#include "EnemyAI.h"
namespace game_framework {
        class Orochimaru : public EnemyAI
        {
        public:
                Orochimaru();
                ~Orochimaru();
                int     GetX1();
                int     GetY1();
                int     GetX2();
                int     GetY2();
                void AttackMovie();
                void AttackMovie2();
                void RunSkillsuccessMovie();
                void Initialize();                      // 設定角色為初始值
                void LoadBitmap();                      // 載入圖形
                void UI(bool Hit);
                void SetpicXY();
                void OnShow();
                bool GetisSkill();
        private:
                int picx, picy;
                int Skilldelay;
                int SasukeDistance;
                int _direction, _distance;
                int SasukeX, SasukeY;
                bool Return;
                bool isSkill2;
        };
}
#endif
```

Orochimaru.cpp

```
#include "stdafx.h"
#include "Resource.h"
#include <mmsystem.h>
#include <ddraw.h>
#include "audio.h"
#include "gamelib.h"
#include "Orochimaru.h"

namespace game_framework {
        Orochimaru::Orochimaru()
```

```
{
	Initialize();
}
Orochimaru::~Orochimaru()
{
}
int Orochimaru::GetX1()
{
	//////////////////////////////////////////////////////圖片 x 座標修正//////////////////////////////////////////////
	if (isSkill || isSkill2) return picx;
	if (direction == 1) return x;
	return x + (StandWidth - animation.Width()) * 2;
	//////////////////////////////////////////////////////////////////////////////////////////////////////////////////
}
int Orochimaru::GetY1()
{
	//////////////////////////////////////////////////////圖片 y 座標修正//////////////////////////////////////////////
	if (isSkill || isSkill2) return picy;
	return y + (StandHeight - animation.Height()) * 2;
	//////////////////////////////////////////////////////////////////////////////////////////////////////////////////
}
int Orochimaru::GetX2()
{
	if (isSkill || isSkill2) return GetX1() + SkillMovie.Width() * 2;
	return GetX1() + animation.Width() * 2;
}
int Orochimaru::GetY2()
{
	if (isSkill || isSkill2) return GetY1() + SkillMovie.Height() * 2;
	return GetY1() + animation.Height() * 2;
}
void Orochimaru::AttackMovie()
{
	//////////////////////////////////////////////////////播放技能動畫//////////////////////////////////////////////
	if (Movie.isStart())
	{
		Movie.OnMove();
		Player->SetMove(false);
		picy = 160;
		if (direction == 0) picx = 980;//設定蛇出來的起始位置
		else picx = -300;
		SkillMovie.Movetonum(1+direction * 2);//蛇咬
	}
	//////////////////////////////////////////////////////技能判定//////////////////////////////////////////////////
	if (!Movie.isStart())
	{
		if (!Skillsuccess)
		{
			if (direction == 0) picx -= 20;
			else picx += 20;
			Player->SetMove(true);
		}
		else
			RunSkillsuccessMovie();
		//////////////////////////////////////////////////超過技能範圍//////////////////////////////////////////////
		if (!Skillsuccess && (direction==0 && picx+SkillMovie.Width()*2<=640) || (direction == 1 && picx >= 0))
		{
			Return = true;
		}
		//////////////////////////////////////////////////回頭//////////////////////////////////////////////////////
		if (!Skillsuccess && Return)
		{
			if (direction == 0) picx += 40;
			else picx -= 40;
		}
		//////////////////////////////////////////////////技能結束//////////////////////////////////////////////////
		if (Return && ((picx == 980 && direction == 0) || (picx == -300 && direction == 1)))
		{
			Move = true;
			isSkill = false;
			Movieflag = false;
			isDie = false;
			isAttacked = false;
			Return = false;
			SkillMovie.Reset();
			Skill = rand() % 10;
```

```
                picx = picy = -400;
            }
        }
        ////////////////////////////////////////////////////////////////////////////////////////////////////
    }
    void Orochimaru::AttackMovie2()
    {
        ////////////////////////////////////////////////////播放技能動畫////////////////////////////////////////////
        if (Movie.isStart())
        {
            Movie.OnMove();
            Player->SetMove(false);
            picy = 160;
            if (direction == 0) picx = 980;//設定蛇出來的起始位置
            else picx = -300;
            SkillMovie.Movetonum(0+direction*2);//蛇撞
        }
        ////////////////////////////////////////////////////技能判定////////////////////////////////////////////////
        if (!Movie.isStart())
        {
            if (direction == 0) picx -= 20;
            else picx += 20;
            Player->SetMove(true);
            ////////////////////////////////////////////////超過技能範圍////////////////////////////////////////////
            if ((direction == 0 && picx + SkillMovie.Width() * 2 <= 640) || (direction == 1 && picx >= 0))
            {
                Return = true;
            }
            ////////////////////////////////////////////////回頭////////////////////////////////////////////////////
            if (Return)
            {
                if (direction == 0) picx += 40;
                else picx -= 40;
            }
            ////////////////////////////////////////////////技能結束////////////////////////////////////////////////
            if (Return && ((picx == 980 && direction == 0) || (picx == -300 && direction == 1)))
            {
                Move = true;
                isSkill2 = false;
                Movieflag = false;
                isDie = false;
                isAttacked = false;
                Return = false;
                Skillsuccess = false;
                SkillMovie.Reset();
                Skill = rand() % 10;
                picx = picy = -400;
            }
            ////////////////////////////////////////////////////////////////////////////////////////////////////
        }
    }
    void Orochimaru::RunSkillsuccessMovie()
    {
        ////////////////////////////////////////////////////技能成功////////////////////////////////////////////////
        if (SkillsuccessMovie.GetCurrentBitmapNumber() == 23)
        {
            Return = true;
            Skillsuccess = false;
            SkillsuccessMovie.Reset();
            Player->SetXY(SasukeX, SasukeY);
            Player->SetDie(2);
        }
        ////////////////////////////////////////////////技能結束回頭////////////////////////////////////////////////
        else if ((direction == 0 && picx + SkillMovie.Width() * 2 >= 640) || (direction == 1 && picx <= 0))
        {
            if (direction == 0) picx -= 20;
            else picx += 20;
        }
        ////////////////////////////////////////////////////////////////////////////////////////////////////
        SkillsuccessMovie.OnMove();
        if (SkillsuccessMovie.GetCurrentBitmapNumber() % 6 == 5) {Player->Attacked(); Player->SubBlood(3);}
        SkillsuccessMovie.SetTopLeft(picx, picy);
    }
    void Orochimaru::Initialize()
    {
        ////////////////////////////////////////////////初始化////////////////////////////////////////////////////
```

```
	floor = 460;
	const int X_POS = 500;
	const int Y_POS = 460 - 39;
	const int INITIAL_VELOCITY = 18;
	initial_velocity = INITIAL_VELOCITY;
	velocity = initial_velocity;
	x = X_POS;
	y = Y_POS;
	isSkill2=Return=isThrow = Movieflag = Skillsuccess = isSkill = Wall = isAttack = isDie = isShooting = isAttacked = rising = isMovingLeft =
	isMovingRight = isMovingUp = false;
	step_size = 10;
	isShow = Move = true;
	attacknum = 0;
	direction = 0;
	StandHeight = 44;
	StandWidth = 22;
	Total = 123, StandStart = 0, StandNum = 8, RunStart = 9, RunNum = 23, JumpStart = 33, JumpNum = 0; AttackStart = 33; AttackNum = 53,
	ThrowStart = 87, ThrowNum = 0, HurtStart = 87, HurtNum = 2, DieStart = 93, DieNum = 29;
	distance = 200;
	Playerdirection = 1;
	Blood = 128;
	delay = 2;
	Skilldelay = 33;
	Skill = rand() % 10;
	Diedelay = 30;
	picx = picy = -400;
	//////////////////////////////////////////////////////////////////////////////////////////////////////////////////
}
void Orochimaru::UI(bool Hit)
{
	//////////////////////////////////////////////////依照對方位置決定面向哪裡//////////////////////////////////////////////////
	if (!Player->GetisMolding() && !Player->GetisSkill() && Hit && !Player->GetisDie() && (GetCurrentBitmapNumber() == 38 +
	GetDirection() * 123 || GetCurrentBitmapNumber() == 53 + GetDirection() * 123 || GetCurrentBitmapNumber() == 74 + GetDirection() * 123))
	{
		Player->SetAttacked(true);
	}
	if (!Player->GetisMolding() && !Player->GetisSkill() && GetisSkill() && Hit && SkillMovie.GetCurrentBitmapNumber() == 1
	&& !GetSkillsuccess() && !Return)
	{
		Player->SetMove(false);
		SasukeX = Player->Getx();
		SasukeY = Player->Gety();
		Player->SetXY(-100, -100);
		SetSkillsuccess();
		GetSkillsuccessMovie()->Movetonum(0 + direction * 24);
	}
	if (!Player->GetisMolding() && !Player->GetisSkill() && GetisSkill() && Hit && SkillMovie.GetCurrentBitmapNumber() == 0 && !Player-
	>GetisDie())
	{
		Player->SetMove(false);
		Player->SetDie(3);
	}
	if (Player->GetX2() < GetX1())
	{
		if (!GetAttack() && !GetAttacked() && !GetDie() && !GetisSkill() && GetAttacknum() == 0)
		{
			_direction = 1;
			SetDirection(0);
		}
	}
	else if (Player->GetX1() > GetX2())
	{
		if (!GetAttack() && !GetAttacked() && !GetDie() && !GetisSkill() && GetAttacknum() == 0)
		{
			_direction = 0;
			SetDirection(1);
		}
	}
	//////////////////////////////////////////////////判斷距離//////////////////////////////////////////////////////////////
	if (!isAttack) Playerdirection = _direction;
	if (Playerdirection == 0) _distance = Player->GetX1() - (x+StandWidth*2);
	else _distance = Player->GetX2() - x;
	_distance = abs(_distance);
	distance = _distance;
	//////////////////////////////////////////////////技能 1 施放//////////////////////////////////////////////////////////
	if (!Player->GetSkillsuccess())
```

```
	{
		if (isAttacked)
		{
			isSkill = false;
			Move = true;
			Movieflag = false;
		}
		else if (Skill == 9 && times <= 0 && Move && distance < 190 || isSkill)
		{
			if (!Movieflag)
			{
				if (Player->GetX1() < GetX2()) direction = 0;
				else if (Player->GetX2() > GetX1()) direction = 1;
				SkillMovie.Movetonum(1);
				Movieflag = true;
				Movie.Init();
				CAudio::Instance()->Play(31);
				picx = picy = -400;
			}
			isMovingRight = false;
			isMovingLeft = false;
			isSkill = true;
			Move = false;
			AttackMovie();
			times = rand() % 75 + 30;
		}
		//////////////////////////////////////////////////////技能 2 施放//////////////////////////////////////////////////////
		else if ((Skill == 8 || Skill == 7) &&times <= 0 && Move && distance < 190 || isSkill2)
		{
			if (!Movieflag)
			{
				if (Player->GetX1() < GetX2()) direction = 0;
				else if (Player->GetX2() > GetX1()) direction = 1;
				SkillMovie.Movetonum(0);
				Movieflag = true;
				Movie.Init();
				CAudio::Instance()->Play(31);
				picx = picy = -400;
			}
			isMovingRight = false;
			isMovingLeft = false;
			isSkill2 = true;
			Move = false;
			AttackMovie2();
			times = rand() % 75 + 30;
		}
		//////////////////////////////////////////////////////攻擊及移動判定//////////////////////////////////////////////////////
		else if (times <= 0 && Move)
		{
			TRACE("%d----------\n", distance);
			if (distance <= 50)
			{
				isAttack = true;
				isMovingRight = false;
				isMovingLeft = false;
				attacknum = 1;
				times = rand() % 30;
				Wall = false;
			}
			else if (distance > 50)
			{
				if (Playerdirection == 1)
				{
					isMovingRight = false;
					isMovingLeft = true;
				}
				else
				{
					isMovingLeft = false;
					isMovingRight = true;
				}
			}
		}
		if (Move) times--;
	}
}
```

```
void Orochimaru::SetpicXY()
{
}
void Orochimaru::OnShow()
{
	///////////////////////////////////////////////顯示/////////////////////////////////////////////////////
	if (!isSkill && !isSkill2)
	{
		animation.SetTopLeft(GetX1(), GetY1());
		animation.OnShow(2);
	}
	else if (!Skillsuccess)
	{
		SkillMovie.SetTopLeft(picx, picy);
		SkillMovie.OnShow(2);
	}
	if (Skillsuccess) SkillsuccessMovie.OnShow(2);
}
bool Orochimaru::GetisSkill()
{
	return isSkill||isSkill2;
}
void Orochimaru::LoadBitmap()
{
	/////////////////////////////////////////////////////////////讀取圖片/////////////////////////////////////////
	animation.AddBitmap(OrochimaruStand1, RGB(255, 255, 255));		//0
	animation.AddBitmap(OrochimaruStand1, RGB(255, 255, 255));		//1
	animation.AddBitmap(OrochimaruStand1, RGB(255, 255, 255));		//2
	animation.AddBitmap(OrochimaruStand2, RGB(255, 255, 255));		//3
	animation.AddBitmap(OrochimaruStand2, RGB(255, 255, 255));		//4
	animation.AddBitmap(OrochimaruStand2, RGB(255, 255, 255));		//5
	animation.AddBitmap(OrochimaruStand3, RGB(255, 255, 255));		//6
	animation.AddBitmap(OrochimaruStand3, RGB(255, 255, 255));		//7
	animation.AddBitmap(OrochimaruStand3, RGB(255, 255, 255));		//8
	animation.AddBitmap(OrochimaruRun1, RGB(255, 255, 255));		//9
	animation.AddBitmap(OrochimaruRun1, RGB(255, 255, 255));		//10
	animation.AddBitmap(OrochimaruRun1, RGB(255, 255, 255));		//1
	animation.AddBitmap(OrochimaruRun2, RGB(255, 255, 255));		//2
	animation.AddBitmap(OrochimaruRun2, RGB(255, 255, 255));		//3
	animation.AddBitmap(OrochimaruRun2, RGB(255, 255, 255));		//4
	animation.AddBitmap(OrochimaruRun3, RGB(255, 255, 255));		//5
	animation.AddBitmap(OrochimaruRun3, RGB(255, 255, 255));		//6
	animation.AddBitmap(OrochimaruRun3, RGB(255, 255, 255));		//7
	animation.AddBitmap(OrochimaruRun4, RGB(255, 255, 255));		//8
	animation.AddBitmap(OrochimaruRun4, RGB(255, 255, 255));		//9
	animation.AddBitmap(OrochimaruRun4, RGB(255, 255, 255));		//20
	animation.AddBitmap(OrochimaruRun5, RGB(255, 255, 255));		//1
	animation.AddBitmap(OrochimaruRun5, RGB(255, 255, 255));		//2
	animation.AddBitmap(OrochimaruRun5, RGB(255, 255, 255));		//3
	animation.AddBitmap(OrochimaruRun6, RGB(255, 255, 255));		//4
	animation.AddBitmap(OrochimaruRun6, RGB(255, 255, 255));		//5
	animation.AddBitmap(OrochimaruRun6, RGB(255, 255, 255));		//6
	animation.AddBitmap(OrochimaruRun7, RGB(255, 255, 255));		//7
	animation.AddBitmap(OrochimaruRun7, RGB(255, 255, 255));		//8
	animation.AddBitmap(OrochimaruRun7, RGB(255, 255, 255));		//9
	animation.AddBitmap(OrochimaruRun8, RGB(255, 255, 255));		//30
	animation.AddBitmap(OrochimaruRun8, RGB(255, 255, 255));		//1
	animation.AddBitmap(OrochimaruRun8, RGB(255, 255, 255));		//2
	animation.AddBitmap(OrochimaruCombo1, RGB(255, 255, 255));		//3
	animation.AddBitmap(OrochimaruCombo1, RGB(255, 255, 255));		//4
	animation.AddBitmap(OrochimaruCombo1, RGB(255, 255, 255));		//5
	animation.AddBitmap(OrochimaruCombo2, RGB(255, 255, 255));		//6
	animation.AddBitmap(OrochimaruCombo2, RGB(255, 255, 255));		//7
	animation.AddBitmap(OrochimaruCombo2, RGB(255, 255, 255));		//8
	animation.AddBitmap(OrochimaruCombo3, RGB(255, 255, 255));		//9
	animation.AddBitmap(OrochimaruCombo3, RGB(255, 255, 255));		//40
	animation.AddBitmap(OrochimaruCombo3, RGB(255, 255, 255));		//1
	animation.AddBitmap(OrochimaruCombo4, RGB(255, 255, 255));		//2
	animation.AddBitmap(OrochimaruCombo4, RGB(255, 255, 255));		//3
	animation.AddBitmap(OrochimaruCombo4, RGB(255, 255, 255));		//4
	animation.AddBitmap(OrochimaruCombo5, RGB(255, 255, 255));		//5
	animation.AddBitmap(OrochimaruCombo5, RGB(255, 255, 255));		//6
	animation.AddBitmap(OrochimaruCombo5, RGB(255, 255, 255));		//7
	animation.AddBitmap(OrochimaruCombo6, RGB(255, 255, 255));		//8
	animation.AddBitmap(OrochimaruCombo6, RGB(255, 255, 255));		//9
	animation.AddBitmap(OrochimaruCombo6, RGB(255, 255, 255));		//50
```

```
animation.AddBitmap(OrochimaruCombo7, RGB(255, 255, 255));          //1
animation.AddBitmap(OrochimaruCombo7, RGB(255, 255, 255));          //2
animation.AddBitmap(OrochimaruCombo7, RGB(255, 255, 255));          //3
animation.AddBitmap(OrochimaruCombo8, RGB(255, 255, 255));          //4
animation.AddBitmap(OrochimaruCombo8, RGB(255, 255, 255));          //5
animation.AddBitmap(OrochimaruCombo8, RGB(255, 255, 255));          //6
animation.AddBitmap(OrochimaruCombo9, RGB(255, 255, 255));          //7
animation.AddBitmap(OrochimaruCombo9, RGB(255, 255, 255));          //8
animation.AddBitmap(OrochimaruCombo9, RGB(255, 255, 255));          //9
animation.AddBitmap(OrochimaruCombo10, RGB(255, 255, 255));                 //60
animation.AddBitmap(OrochimaruCombo10, RGB(255, 255, 255));                 //1
animation.AddBitmap(OrochimaruCombo10, RGB(255, 255, 255));                 //2
animation.AddBitmap(OrochimaruCombo11, RGB(255, 255, 255));                 //3
animation.AddBitmap(OrochimaruCombo11, RGB(255, 255, 255));                 //4
animation.AddBitmap(OrochimaruCombo11, RGB(255, 255, 255));                 //5
animation.AddBitmap(OrochimaruCombo12, RGB(255, 255, 255));                 //6
animation.AddBitmap(OrochimaruCombo12, RGB(255, 255, 255));                 //7
animation.AddBitmap(OrochimaruCombo12, RGB(255, 255, 255));                 //8
animation.AddBitmap(OrochimaruCombo13, RGB(255, 255, 255));                 //9
animation.AddBitmap(OrochimaruCombo13, RGB(255, 255, 255));                 //70
animation.AddBitmap(OrochimaruCombo13, RGB(255, 255, 255));                 //1
animation.AddBitmap(OrochimaruCombo14, RGB(255, 255, 255));                 //2
animation.AddBitmap(OrochimaruCombo14, RGB(255, 255, 255));                 //3
animation.AddBitmap(OrochimaruCombo14, RGB(255, 255, 255));                 //4
animation.AddBitmap(OrochimaruCombo15, RGB(255, 255, 255));                 //5
animation.AddBitmap(OrochimaruCombo15, RGB(255, 255, 255));                 //6
animation.AddBitmap(OrochimaruCombo15, RGB(255, 255, 255));                 //7
animation.AddBitmap(OrochimaruCombo16, RGB(255, 255, 255));                 //8
animation.AddBitmap(OrochimaruCombo16, RGB(255, 255, 255));                 //9
animation.AddBitmap(OrochimaruCombo16, RGB(255, 255, 255));                 //80
animation.AddBitmap(OrochimaruCombo17, RGB(255, 255, 255));                 //1
animation.AddBitmap(OrochimaruCombo17, RGB(255, 255, 255));                 //2
animation.AddBitmap(OrochimaruCombo17, RGB(255, 255, 255));                 //3
animation.AddBitmap(OrochimaruCombo18, RGB(255, 255, 255));                 //4
animation.AddBitmap(OrochimaruCombo18, RGB(255, 255, 255));                 //5
animation.AddBitmap(OrochimaruCombo18, RGB(255, 255, 255));                 //6
animation.AddBitmap(OrochimaruHurt, RGB(255, 255, 255));            //7
animation.AddBitmap(OrochimaruHurt, RGB(255, 255, 255));            //8
animation.AddBitmap(OrochimaruHurt, RGB(255, 255, 255));            //9
animation.AddBitmap(OrochimaruHurt, RGB(255, 255, 255));            //90
animation.AddBitmap(OrochimaruHurt, RGB(255, 255, 255));            //1
animation.AddBitmap(OrochimaruHurt, RGB(255, 255, 255));            //2
animation.AddBitmap(OrochimaruHurt, RGB(255, 255, 255));            //3
animation.AddBitmap(OrochimaruHurt, RGB(255, 255, 255));            //4
animation.AddBitmap(OrochimaruHurt, RGB(255, 255, 255));            //5
animation.AddBitmap(OrochimaruDie1, RGB(255, 255, 255));            //6
animation.AddBitmap(OrochimaruDie1, RGB(255, 255, 255));            //7
animation.AddBitmap(OrochimaruDie1, RGB(255, 255, 255));            //8
animation.AddBitmap(OrochimaruDie2, RGB(255, 255, 255));            //9
animation.AddBitmap(OrochimaruDie2, RGB(255, 255, 255));            //100
animation.AddBitmap(OrochimaruDie2, RGB(255, 255, 255));            //1
animation.AddBitmap(OrochimaruDie2, RGB(255, 255, 255));            //2
animation.AddBitmap(OrochimaruDie2, RGB(255, 255, 255));            //3
animation.AddBitmap(OrochimaruDie2, RGB(255, 255, 255));            //4
animation.AddBitmap(OrochimaruDie3, RGB(255, 255, 255));            //5
animation.AddBitmap(OrochimaruDie3, RGB(255, 255, 255));            //6
animation.AddBitmap(OrochimaruDie3, RGB(255, 255, 255));            //7
animation.AddBitmap(OrochimaruDie3, RGB(255, 255, 255));            //8
animation.AddBitmap(OrochimaruDie3, RGB(255, 255, 255));            //9
animation.AddBitmap(OrochimaruDie3, RGB(255, 255, 255));            //110
animation.AddBitmap(OrochimaruDie4, RGB(255, 255, 255));            //1
animation.AddBitmap(OrochimaruDie4, RGB(255, 255, 255));            //2
animation.AddBitmap(OrochimaruDie4, RGB(255, 255, 255));            //3
animation.AddBitmap(OrochimaruDie4, RGB(255, 255, 255));            //4
animation.AddBitmap(OrochimaruDie4, RGB(255, 255, 255));            //5
animation.AddBitmap(OrochimaruDie4, RGB(255, 255, 255));            //6
animation.AddBitmap(OrochimaruDie4, RGB(255, 255, 255));            //7
animation.AddBitmap(OrochimaruDie4, RGB(255, 255, 255));            //8
animation.AddBitmap(OrochimaruDie4, RGB(255, 255, 255));            //9
animation.AddBitmap(OrochimaruDie4, RGB(255, 255, 255));            //120
animation.AddBitmap(OrochimaruDie4, RGB(255, 255, 255));            //1
animation.AddBitmap(OrochimaruDie4, RGB(255, 255, 255));            //2
animation.AddBitmap(OrochimaruStand4, RGB(255, 255, 255));          //3
animation.AddBitmap(OrochimaruStand4, RGB(255, 255, 255));          //4
animation.AddBitmap(OrochimaruStand4, RGB(255, 255, 255));          //5
animation.AddBitmap(OrochimaruStand5, RGB(255, 255, 255));          //6
```

```
animation.AddBitmap(OrochimaruStand5, RGB(255, 255, 255));          //7
animation.AddBitmap(OrochimaruStand5, RGB(255, 255, 255));          //8
animation.AddBitmap(OrochimaruStand6, RGB(255, 255, 255));          //9
animation.AddBitmap(OrochimaruStand6, RGB(255, 255, 255));          //130
animation.AddBitmap(OrochimaruStand6, RGB(255, 255, 255));          //1
animation.AddBitmap(OrochimaruRun9, RGB(255, 255, 255));            //2
animation.AddBitmap(OrochimaruRun9, RGB(255, 255, 255));            //3
animation.AddBitmap(OrochimaruRun9, RGB(255, 255, 255));            //4
animation.AddBitmap(OrochimaruRun10, RGB(255, 255, 255));           //5
animation.AddBitmap(OrochimaruRun10, RGB(255, 255, 255));           //6
animation.AddBitmap(OrochimaruRun10, RGB(255, 255, 255));           //7
animation.AddBitmap(OrochimaruRun11, RGB(255, 255, 255));           //8
animation.AddBitmap(OrochimaruRun11, RGB(255, 255, 255));           //9
animation.AddBitmap(OrochimaruRun11, RGB(255, 255, 255));           //140
animation.AddBitmap(OrochimaruRun12, RGB(255, 255, 255));           //1
animation.AddBitmap(OrochimaruRun12, RGB(255, 255, 255));           //2
animation.AddBitmap(OrochimaruRun12, RGB(255, 255, 255));           //3
animation.AddBitmap(OrochimaruRun13, RGB(255, 255, 255));           //4
animation.AddBitmap(OrochimaruRun13, RGB(255, 255, 255));           //5
animation.AddBitmap(OrochimaruRun13, RGB(255, 255, 255));           //6
animation.AddBitmap(OrochimaruRun14, RGB(255, 255, 255));           //7
animation.AddBitmap(OrochimaruRun14, RGB(255, 255, 255));           //8
animation.AddBitmap(OrochimaruRun14, RGB(255, 255, 255));           //9
animation.AddBitmap(OrochimaruRun15, RGB(255, 255, 255));           //150
animation.AddBitmap(OrochimaruRun15, RGB(255, 255, 255));           //1
animation.AddBitmap(OrochimaruRun15, RGB(255, 255, 255));           //2
animation.AddBitmap(OrochimaruRun16, RGB(255, 255, 255));           //3
animation.AddBitmap(OrochimaruRun16, RGB(255, 255, 255));           //4
animation.AddBitmap(OrochimaruRun16, RGB(255, 255, 255));           //5
animation.AddBitmap(OrochimaruCombo19, RGB(255, 255, 255));             //6
animation.AddBitmap(OrochimaruCombo19, RGB(255, 255, 255));             //7
animation.AddBitmap(OrochimaruCombo19, RGB(255, 255, 255));             //8
animation.AddBitmap(OrochimaruCombo20, RGB(255, 255, 255));             //9
animation.AddBitmap(OrochimaruCombo20, RGB(255, 255, 255));             //160
animation.AddBitmap(OrochimaruCombo20, RGB(255, 255, 255));             //1
animation.AddBitmap(OrochimaruCombo21, RGB(255, 255, 255));             //2
animation.AddBitmap(OrochimaruCombo21, RGB(255, 255, 255));             //3
animation.AddBitmap(OrochimaruCombo21, RGB(255, 255, 255));             //4
animation.AddBitmap(OrochimaruCombo22, RGB(255, 255, 255));             //5
animation.AddBitmap(OrochimaruCombo22, RGB(255, 255, 255));             //6
animation.AddBitmap(OrochimaruCombo22, RGB(255, 255, 255));             //7
animation.AddBitmap(OrochimaruCombo23, RGB(255, 255, 255));             //8
animation.AddBitmap(OrochimaruCombo23, RGB(255, 255, 255));             //9
animation.AddBitmap(OrochimaruCombo23, RGB(255, 255, 255));             //170
animation.AddBitmap(OrochimaruCombo24, RGB(255, 255, 255));             //1
animation.AddBitmap(OrochimaruCombo24, RGB(255, 255, 255));             //2
animation.AddBitmap(OrochimaruCombo24, RGB(255, 255, 255));             //3
animation.AddBitmap(OrochimaruCombo25, RGB(255, 255, 255));             //4
animation.AddBitmap(OrochimaruCombo25, RGB(255, 255, 255));             //5
animation.AddBitmap(OrochimaruCombo25, RGB(255, 255, 255));             //6
animation.AddBitmap(OrochimaruCombo26, RGB(255, 255, 255));             //7
animation.AddBitmap(OrochimaruCombo26, RGB(255, 255, 255));             //8
animation.AddBitmap(OrochimaruCombo26, RGB(255, 255, 255));             //9
animation.AddBitmap(OrochimaruCombo27, RGB(255, 255, 255));             //180
animation.AddBitmap(OrochimaruCombo27, RGB(255, 255, 255));             //1
animation.AddBitmap(OrochimaruCombo27, RGB(255, 255, 255));             //2
animation.AddBitmap(OrochimaruCombo28, RGB(255, 255, 255));             //3
animation.AddBitmap(OrochimaruCombo28, RGB(255, 255, 255));             //4
animation.AddBitmap(OrochimaruCombo28, RGB(255, 255, 255));             //5
animation.AddBitmap(OrochimaruCombo29, RGB(255, 255, 255));             //6
animation.AddBitmap(OrochimaruCombo29, RGB(255, 255, 255));             //7
animation.AddBitmap(OrochimaruCombo29, RGB(255, 255, 255));             //8
animation.AddBitmap(OrochimaruCombo30, RGB(255, 255, 255));             //9
animation.AddBitmap(OrochimaruCombo30, RGB(255, 255, 255));             //190
animation.AddBitmap(OrochimaruCombo30, RGB(255, 255, 255));             //1
animation.AddBitmap(OrochimaruCombo31, RGB(255, 255, 255));             //2
animation.AddBitmap(OrochimaruCombo31, RGB(255, 255, 255));             //3
animation.AddBitmap(OrochimaruCombo31, RGB(255, 255, 255));             //4
animation.AddBitmap(OrochimaruCombo32, RGB(255, 255, 255));             //5
animation.AddBitmap(OrochimaruCombo32, RGB(255, 255, 255));             //6
animation.AddBitmap(OrochimaruCombo32, RGB(255, 255, 255));             //7
animation.AddBitmap(OrochimaruCombo33, RGB(255, 255, 255));             //8
animation.AddBitmap(OrochimaruCombo33, RGB(255, 255, 255));             //9
animation.AddBitmap(OrochimaruCombo33, RGB(255, 255, 255));             //200
animation.AddBitmap(OrochimaruCombo34, RGB(255, 255, 255));             //1
animation.AddBitmap(OrochimaruCombo34, RGB(255, 255, 255));             //2
```

```
animation.AddBitmap(OrochimaruCombo34, RGB(255, 255, 255));            //3
animation.AddBitmap(OrochimaruCombo35, RGB(255, 255, 255));            //4
animation.AddBitmap(OrochimaruCombo35, RGB(255, 255, 255));            //5
animation.AddBitmap(OrochimaruCombo35, RGB(255, 255, 255));            //6
animation.AddBitmap(OrochimaruCombo36, RGB(255, 255, 255));            //7
animation.AddBitmap(OrochimaruCombo36, RGB(255, 255, 255));            //8
animation.AddBitmap(OrochimaruCombo36, RGB(255, 255, 255));            //9
animation.AddBitmap(OrochimaruHurt2, RGB(255, 255, 255));        //210
animation.AddBitmap(OrochimaruHurt2, RGB(255, 255, 255));        //1
animation.AddBitmap(OrochimaruHurt2, RGB(255, 255, 255));        //2
animation.AddBitmap(OrochimaruHurt2, RGB(255, 255, 255));        //3
animation.AddBitmap(OrochimaruHurt2, RGB(255, 255, 255));        //4
animation.AddBitmap(OrochimaruHurt2, RGB(255, 255, 255));        //5
animation.AddBitmap(OrochimaruHurt2, RGB(255, 255, 255));        //6
animation.AddBitmap(OrochimaruHurt2, RGB(255, 255, 255));        //7
animation.AddBitmap(OrochimaruHurt2, RGB(255, 255, 255));        //8
animation.AddBitmap(OrochimaruDie5, RGB(255, 255, 255));         //9
animation.AddBitmap(OrochimaruDie5, RGB(255, 255, 255));         //220
animation.AddBitmap(OrochimaruDie5, RGB(255, 255, 255));         //1
animation.AddBitmap(OrochimaruDie6, RGB(255, 255, 255));         //2
animation.AddBitmap(OrochimaruDie6, RGB(255, 255, 255));         //3
animation.AddBitmap(OrochimaruDie6, RGB(255, 255, 255));         //4
animation.AddBitmap(OrochimaruDie6, RGB(255, 255, 255));         //5
animation.AddBitmap(OrochimaruDie6, RGB(255, 255, 255));         //6
animation.AddBitmap(OrochimaruDie6, RGB(255, 255, 255));         //7
animation.AddBitmap(OrochimaruDie7, RGB(255, 255, 255));         //8
animation.AddBitmap(OrochimaruDie7, RGB(255, 255, 255));         //9
animation.AddBitmap(OrochimaruDie7, RGB(255, 255, 255));         //230
animation.AddBitmap(OrochimaruDie7, RGB(255, 255, 255));         //1
animation.AddBitmap(OrochimaruDie7, RGB(255, 255, 255));         //2
animation.AddBitmap(OrochimaruDie7, RGB(255, 255, 255));         //3
animation.AddBitmap(OrochimaruDie8, RGB(255, 255, 255));         //4
animation.AddBitmap(OrochimaruDie8, RGB(255, 255, 255));         //5
animation.AddBitmap(OrochimaruDie8, RGB(255, 255, 255));         //6
animation.AddBitmap(OrochimaruDie8, RGB(255, 255, 255));         //7
animation.AddBitmap(OrochimaruDie8, RGB(255, 255, 255));         //8
animation.AddBitmap(OrochimaruDie8, RGB(255, 255, 255));         //9
animation.AddBitmap(OrochimaruDie8, RGB(255, 255, 255));         //240
animation.AddBitmap(OrochimaruDie8, RGB(255, 255, 255));         //1
animation.AddBitmap(OrochimaruDie8, RGB(255, 255, 255));         //2
animation.AddBitmap(OrochimaruDie8, RGB(255, 255, 255));         //3
animation.AddBitmap(OrochimaruDie8, RGB(255, 255, 255));         //4
animation.AddBitmap(OrochimaruDie8, RGB(255, 255, 255));         //5
SkillMovie.AddBitmap(OrochimaruSkill1, RGB(255, 255, 255));
SkillMovie.AddBitmap(OrochimaruSkill2, RGB(255, 255, 255));
SkillMovie.AddBitmap(OrochimaruSkill5, RGB(255, 255, 255));
SkillMovie.AddBitmap(OrochimaruSkill6, RGB(255, 255, 255));
SkillsuccessMovie.AddBitmap(OrochimaruSkill3, RGB(255, 255, 255));
SkillsuccessMovie.AddBitmap(OrochimaruSkill3, RGB(255, 255, 255));
SkillsuccessMovie.AddBitmap(OrochimaruSkill3, RGB(255, 255, 255));
SkillsuccessMovie.AddBitmap(OrochimaruSkill4, RGB(255, 255, 255));
SkillsuccessMovie.AddBitmap(OrochimaruSkill4, RGB(255, 255, 255));
SkillsuccessMovie.AddBitmap(OrochimaruSkill4, RGB(255, 255, 255));
SkillsuccessMovie.AddBitmap(OrochimaruSkill3, RGB(255, 255, 255));
SkillsuccessMovie.AddBitmap(OrochimaruSkill3, RGB(255, 255, 255));
SkillsuccessMovie.AddBitmap(OrochimaruSkill3, RGB(255, 255, 255));
SkillsuccessMovie.AddBitmap(OrochimaruSkill4, RGB(255, 255, 255));
SkillsuccessMovie.AddBitmap(OrochimaruSkill4, RGB(255, 255, 255));
SkillsuccessMovie.AddBitmap(OrochimaruSkill4, RGB(255, 255, 255));
SkillsuccessMovie.AddBitmap(OrochimaruSkill3, RGB(255, 255, 255));
SkillsuccessMovie.AddBitmap(OrochimaruSkill3, RGB(255, 255, 255));
SkillsuccessMovie.AddBitmap(OrochimaruSkill3, RGB(255, 255, 255));
SkillsuccessMovie.AddBitmap(OrochimaruSkill4, RGB(255, 255, 255));
SkillsuccessMovie.AddBitmap(OrochimaruSkill4, RGB(255, 255, 255));
SkillsuccessMovie.AddBitmap(OrochimaruSkill4, RGB(255, 255, 255));
SkillsuccessMovie.AddBitmap(OrochimaruSkill3, RGB(255, 255, 255));
SkillsuccessMovie.AddBitmap(OrochimaruSkill3, RGB(255, 255, 255));
SkillsuccessMovie.AddBitmap(OrochimaruSkill3, RGB(255, 255, 255));
SkillsuccessMovie.AddBitmap(OrochimaruSkill4, RGB(255, 255, 255));
SkillsuccessMovie.AddBitmap(OrochimaruSkill4, RGB(255, 255, 255));
SkillsuccessMovie.AddBitmap(OrochimaruSkill4, RGB(255, 255, 255));
SkillsuccessMovie.AddBitmap(OrochimaruSkill7, RGB(255, 255, 255));
SkillsuccessMovie.AddBitmap(OrochimaruSkill7, RGB(255, 255, 255));
SkillsuccessMovie.AddBitmap(OrochimaruSkill7, RGB(255, 255, 255));
SkillsuccessMovie.AddBitmap(OrochimaruSkill8, RGB(255, 255, 255));
SkillsuccessMovie.AddBitmap(OrochimaruSkill8, RGB(255, 255, 255));
```

```
			SkillsuccessMovie.AddBitmap(OrochimaruSkill8, RGB(255, 255, 255));
			SkillsuccessMovie.AddBitmap(OrochimaruSkill7, RGB(255, 255, 255));
			SkillsuccessMovie.AddBitmap(OrochimaruSkill7, RGB(255, 255, 255));
			SkillsuccessMovie.AddBitmap(OrochimaruSkill7, RGB(255, 255, 255));
			SkillsuccessMovie.AddBitmap(OrochimaruSkill8, RGB(255, 255, 255));
			SkillsuccessMovie.AddBitmap(OrochimaruSkill8, RGB(255, 255, 255));
			SkillsuccessMovie.AddBitmap(OrochimaruSkill8, RGB(255, 255, 255));
			SkillsuccessMovie.AddBitmap(OrochimaruSkill7, RGB(255, 255, 255));
			SkillsuccessMovie.AddBitmap(OrochimaruSkill7, RGB(255, 255, 255));
			SkillsuccessMovie.AddBitmap(OrochimaruSkill7, RGB(255, 255, 255));
			SkillsuccessMovie.AddBitmap(OrochimaruSkill8, RGB(255, 255, 255));
			SkillsuccessMovie.AddBitmap(OrochimaruSkill8, RGB(255, 255, 255));
			SkillsuccessMovie.AddBitmap(OrochimaruSkill8, RGB(255, 255, 255));
			SkillsuccessMovie.AddBitmap(OrochimaruSkill7, RGB(255, 255, 255));
			SkillsuccessMovie.AddBitmap(OrochimaruSkill7, RGB(255, 255, 255));
			SkillsuccessMovie.AddBitmap(OrochimaruSkill7, RGB(255, 255, 255));
			SkillsuccessMovie.AddBitmap(OrochimaruSkill8, RGB(255, 255, 255));
			SkillsuccessMovie.AddBitmap(OrochimaruSkill8, RGB(255, 255, 255));
			SkillsuccessMovie.AddBitmap(OrochimaruSkill8, RGB(255, 255, 255));
			Movie.LoadBitmap(OrochimaruSkillUI, RGB(255, 255, 255));
			SkillMovie.SetDelayCount(1);
			SkillMovie.Reset();
			SkillsuccessMovie.SetDelayCount(1);
			SkillsuccessMovie.Reset();
		}
}
```

Shoot.h

```
#ifndef SHOOT_H
#define SHOOT_H
namespace game_framework {	//繼承用//
	class Shoot
	{
	public:
		Shoot();
		virtual void Init(int _x ,int _y ,int _direction)=0;
		virtual int GetX1()=0;				// 角色左上角 x 座標
		virtual int GetY1()=0;				// 角色左上角 y 座標
		virtual int GetX2()=0;
		virtual int GetY2()=0;
		virtual int Getdirection()=0;
		virtual void AddPic()=0;			// 載入圖形
		virtual void OnShow()=0;
		virtual void Move()=0;
		virtual void SetXY(int _x, int _y)=0;
	protected:
		CAnimation Shurikanpic,Fireballpic;	// 利用動畫作圖形
		int x, y, direction;
		string characteristic;
	};
}
#endif
```

Shoot.cpp

```
#include "stdafx.h"
#include "Resource.h"
#include <mmsystem.h>
#include <ddraw.h>
#include "audio.h"
#include "gamelib.h"
#include "Shoot.h"
namespace game_framework {
	Shoot::Shoot()
	{
	}
}
```

Kakashi.h

```
#ifndef KAKASHI_H
#define KAKASHI_H
namespace game_framework {
	class Kakashi
	{
	public:
		Kakashi();
		void Init();
		void OnShow(int _section);
		void ChangeFilm(int _film);
		int GetFilm();
```

```
		void ChangeVicFilm(int _film);
		void ChangeVic2Film(int _film);
		int GetVicFilm();
		int GetVic2Film();
		int GetSection();
		void ChangeSection(int _section);
		void AddPic();
		bool GetEnd();
		bool IsFinish();
		bool IsDone();
		void SetDone(bool _flag);
	protected:
		CAnimation instructions;
		CAnimation victory;
		CAnimation failed;
		CAnimation victory2;
		CAnimation final;
		bool isEnd, Done, isFinish;
		int nowFilm, nowVicFilm, section, nowVic2Film;
		int film[7] = {18,29,43,60,75,85,102};                    //紀錄每一個對話框使用幾張圖
		int vicfilm[2] = {13,24};                                 //紀錄每一個對話框使用幾張圖
		int vic2film[2] = { 13,30 };                              //紀錄每一個對話框使用幾張圖
	};
}
#endif
```

Kakashi.cpp

```
#include "stdafx.h"
#include "Resource.h"
#include <mmsystem.h>
#include <ddraw.h>
#include "audio.h"
#include "gamelib.h"
#include "Kakashi.h"

namespace game_framework {
	Kakashi::Kakashi()
	{

	}
	void Kakashi::Init()
	{
		AddPic();
		instructions.Reset();
		instructions.SetDelayCount(5);
		instructions.SetTopLeft(0, 55);
		victory.Reset();
		victory.SetDelayCount(5);
		victory.SetTopLeft(0, 55);
		victory2.Reset();
		victory2.SetDelayCount(5);
		victory2.SetTopLeft(0, 55);
		failed.Reset();
		failed.SetDelayCount(5);
		failed.SetTopLeft(0, 55);
		final.Reset();
		final.SetDelayCount(5);
		final.SetTopLeft(0, 55);
		isEnd = false;
		Done = false;
		isFinish = false;
		section = 1;
		nowVicFilm = 0;
		nowVic2Film = 0;
	}

	void Kakashi::OnShow(int _section)
	{
		////////////////////////////判斷目前要顯示的對話場景//////////////////////////
		if (_section == 1) {
			if (Done == false)instructions.OnShow();
			if (instructions.GetCurrentBitmapNumber() < film[nowFilm] - 1 && isFinish == false && Done == false && isEnd == false)
			instructions.Movetonum(instructions.GetCurrentBitmapNumber() + 1);
			if (instructions.GetCurrentBitmapNumber() == film[nowFilm] - 1 && isFinish == false && Done == false && isEnd == false)isEnd =
			true;
			if (instructions.IsFinalBitmap())isFinish = true;
		}
```

```
	else if (_section == 2) {
		if (Done == false)victory.OnShow();
		if (victory.GetCurrentBitmapNumber() < vicfilm[nowVicFilm] - 1 && isFinish == false && Done == false && isEnd == false)
		victory.Movetonum(victory.GetCurrentBitmapNumber() + 1);
		if (victory.GetCurrentBitmapNumber() == vicfilm[nowVicFilm] - 1 && isFinish == false && Done == false && isEnd == false)isEnd =
		true;
		if (victory.IsFinalBitmap())isFinish = true;
	}
	else if (_section == 3) {
		if (Done == false)failed.OnShow();
		if (failed.GetCurrentBitmapNumber() < 10 && isFinish == false && Done == false && isEnd == false)
		failed.Movetonum(failed.GetCurrentBitmapNumber() + 1);
		if (failed.GetCurrentBitmapNumber() == 10 && isFinish == false && Done == false && isEnd == false)isEnd = true;
		if (failed.IsFinalBitmap())isFinish = true;
	}
	else if (_section == 4) {
		if (Done == false)victory2.OnShow();
		if (victory2.GetCurrentBitmapNumber() < vic2film[nowVic2Film] - 1 && isFinish == false && Done == false && isEnd== false)
		victory2.Movetonum(victory2.GetCurrentBitmapNumber() + 1);
		if (victory2.GetCurrentBitmapNumber() == vic2film[nowVic2Film] - 1 && isFinish == false && Done == false && isEnd ==
		false)isEnd = true;
		if (victory2.IsFinalBitmap())isFinish = true;
	}
	else if (_section == 5) {
		if (Done == false)final.OnShow();
		if (final.GetCurrentBitmapNumber() < 10 && isFinish == false && Done == false && isEnd == false)
		final.Movetonum(final.GetCurrentBitmapNumber() + 1);
		if (final.GetCurrentBitmapNumber() == 10 && isFinish == false && Done == false && isEnd == false)isEnd = true;
		if (final.IsFinalBitmap())isFinish = true;
	}
	//////////////////////////////////////////////////////////////////////////
}
void Kakashi::ChangeFilm(int _film)			//切換對話框
{
	if(_film < 7)nowFilm = _film;
	isEnd = false;
}
int Kakashi::GetFilm()
{
	return nowFilm;
}
void Kakashi::ChangeVicFilm(int _film)			//切換對話框
{
	if(_film<2)nowVicFilm = _film;
	isEnd = false;
}
void Kakashi::ChangeVic2Film(int _film)			//切換對話框
{
	if (_film < 2)nowVic2Film = _film;
	isEnd = false;
}
int Kakashi::GetVicFilm()
{
	return nowVicFilm;
}
int Kakashi::GetVic2Film()
{
	return nowVic2Film;
}
int Kakashi::GetSection()
{
	return section;
}
void Kakashi::ChangeSection(int _section)		//切換場景
{
	isEnd = false;
	Done = false;
	isFinish = false;
	victory.Reset();
	victory2.Reset();
	instructions.Reset();
	failed.Reset();
	final.Reset();
	nowVicFilm = 0;
	nowVic2Film = 0;
	nowFilm = 0;
```

```
	section = _section;
}
void Kakashi::SetDone(bool _flag)
{
	Done = _flag;
}
void Kakashi::AddPic()
{
	instructions.AddBitmap(KAKASHI001, RGB(255, 242, 0));
	instructions.AddBitmap(KAKASHI002, RGB(255, 242, 0));
	instructions.AddBitmap(KAKASHI003, RGB(255, 242, 0));
	instructions.AddBitmap(KAKASHI004, RGB(255, 242, 0));
	instructions.AddBitmap(KAKASHI005, RGB(255, 242, 0));
	instructions.AddBitmap(KAKASHI006, RGB(255, 242, 0));
	instructions.AddBitmap(KAKASHI007, RGB(255, 242, 0));
	instructions.AddBitmap(KAKASHI008, RGB(255, 242, 0));
	instructions.AddBitmap(KAKASHI009, RGB(255, 242, 0));
	instructions.AddBitmap(KAKASHI010, RGB(255, 242, 0));
	instructions.AddBitmap(KAKASHI011, RGB(255, 242, 0));
	instructions.AddBitmap(KAKASHI012, RGB(255, 242, 0));
	instructions.AddBitmap(KAKASHI013, RGB(255, 242, 0));
	instructions.AddBitmap(KAKASHI014, RGB(255, 242, 0));
	instructions.AddBitmap(KAKASHI015, RGB(255, 242, 0));
	instructions.AddBitmap(KAKASHI016, RGB(255, 242, 0));
	instructions.AddBitmap(KAKASHI017, RGB(255, 242, 0));
	instructions.AddBitmap(KAKASHI018, RGB(255, 242, 0));
	instructions.AddBitmap(KAKASHI101, RGB(255, 242, 0));
	instructions.AddBitmap(KAKASHI102, RGB(255, 242, 0));
	instructions.AddBitmap(KAKASHI103, RGB(255, 242, 0));
	instructions.AddBitmap(KAKASHI104, RGB(255, 242, 0));
	instructions.AddBitmap(KAKASHI105, RGB(255, 242, 0));
	instructions.AddBitmap(KAKASHI106, RGB(255, 242, 0));
	instructions.AddBitmap(KAKASHI107, RGB(255, 242, 0));
	instructions.AddBitmap(KAKASHI108, RGB(255, 242, 0));
	instructions.AddBitmap(KAKASHI109, RGB(255, 242, 0));
	instructions.AddBitmap(KAKASHI110, RGB(255, 242, 0));
	instructions.AddBitmap(KAKASHI111, RGB(255, 242, 0));
	instructions.AddBitmap(KAKASHI201, RGB(255, 242, 0));
	instructions.AddBitmap(KAKASHI202, RGB(255, 242, 0));
	instructions.AddBitmap(KAKASHI203, RGB(255, 242, 0));
	instructions.AddBitmap(KAKASHI204, RGB(255, 242, 0));
	instructions.AddBitmap(KAKASHI205, RGB(255, 242, 0));
	instructions.AddBitmap(KAKASHI206, RGB(255, 242, 0));
	instructions.AddBitmap(KAKASHI207, RGB(255, 242, 0));
	instructions.AddBitmap(KAKASHI208, RGB(255, 242, 0));
	instructions.AddBitmap(KAKASHI209, RGB(255, 242, 0));
	instructions.AddBitmap(KAKASHI210, RGB(255, 242, 0));
	instructions.AddBitmap(KAKASHI211, RGB(255, 242, 0));
	instructions.AddBitmap(KAKASHI212, RGB(255, 242, 0));
	instructions.AddBitmap(KAKASHI213, RGB(255, 242, 0));
	instructions.AddBitmap(KAKASHI214, RGB(255, 242, 0));
	instructions.AddBitmap(KAKASHI301, RGB(255, 242, 0));
	instructions.AddBitmap(KAKASHI302, RGB(255, 242, 0));
	instructions.AddBitmap(KAKASHI303, RGB(255, 242, 0));
	instructions.AddBitmap(KAKASHI304, RGB(255, 242, 0));
	instructions.AddBitmap(KAKASHI305, RGB(255, 242, 0));
	instructions.AddBitmap(KAKASHI306, RGB(255, 242, 0));
	instructions.AddBitmap(KAKASHI307, RGB(255, 242, 0));
	instructions.AddBitmap(KAKASHI308, RGB(255, 242, 0));
	instructions.AddBitmap(KAKASHI309, RGB(255, 242, 0));
	instructions.AddBitmap(KAKASHI310, RGB(255, 242, 0));
	instructions.AddBitmap(KAKASHI311, RGB(255, 242, 0));
	instructions.AddBitmap(KAKASHI312, RGB(255, 242, 0));
	instructions.AddBitmap(KAKASHI313, RGB(255, 242, 0));
	instructions.AddBitmap(KAKASHI314, RGB(255, 242, 0));
	instructions.AddBitmap(KAKASHI315, RGB(255, 242, 0));
	instructions.AddBitmap(KAKASHI316, RGB(255, 242, 0));
	instructions.AddBitmap(KAKASHI317, RGB(255, 242, 0));
	instructions.AddBitmap(KAKASHI401, RGB(255, 242, 0));
	instructions.AddBitmap(KAKASHI402, RGB(255, 242, 0));
	instructions.AddBitmap(KAKASHI403, RGB(255, 242, 0));
	instructions.AddBitmap(KAKASHI404, RGB(255, 242, 0));
	instructions.AddBitmap(KAKASHI405, RGB(255, 242, 0));
	instructions.AddBitmap(KAKASHI406, RGB(255, 242, 0));
	instructions.AddBitmap(KAKASHI407, RGB(255, 242, 0));
	instructions.AddBitmap(KAKASHI408, RGB(255, 242, 0));
```

```
instructions.AddBitmap(KAKASHI409, RGB(255, 242, 0));
instructions.AddBitmap(KAKASHI410, RGB(255, 242, 0));
instructions.AddBitmap(KAKASHI411, RGB(255, 242, 0));
instructions.AddBitmap(KAKASHI412, RGB(255, 242, 0));
instructions.AddBitmap(KAKASHI413, RGB(255, 242, 0));
instructions.AddBitmap(KAKASHI414, RGB(255, 242, 0));
instructions.AddBitmap(KAKASHI415, RGB(255, 242, 0));
instructions.AddBitmap(KAKASHI501, RGB(255, 242, 0));
instructions.AddBitmap(KAKASHI502, RGB(255, 242, 0));
instructions.AddBitmap(KAKASHI503, RGB(255, 242, 0));
instructions.AddBitmap(KAKASHI504, RGB(255, 242, 0));
instructions.AddBitmap(KAKASHI505, RGB(255, 242, 0));
instructions.AddBitmap(KAKASHI506, RGB(255, 242, 0));
instructions.AddBitmap(KAKASHI507, RGB(255, 242, 0));
instructions.AddBitmap(KAKASHI508, RGB(255, 242, 0));
instructions.AddBitmap(KAKASHI509, RGB(255, 242, 0));
instructions.AddBitmap(KAKASHI510, RGB(255, 242, 0));
instructions.AddBitmap(KAKASHI601, RGB(255, 242, 0));
instructions.AddBitmap(KAKASHI602, RGB(255, 242, 0));
instructions.AddBitmap(KAKASHI603, RGB(255, 242, 0));
instructions.AddBitmap(KAKASHI604, RGB(255, 242, 0));
instructions.AddBitmap(KAKASHI605, RGB(255, 242, 0));
instructions.AddBitmap(KAKASHI606, RGB(255, 242, 0));
instructions.AddBitmap(KAKASHI607, RGB(255, 242, 0));
instructions.AddBitmap(KAKASHI608, RGB(255, 242, 0));
instructions.AddBitmap(KAKASHI609, RGB(255, 242, 0));
instructions.AddBitmap(KAKASHI610, RGB(255, 242, 0));
instructions.AddBitmap(KAKASHI611, RGB(255, 242, 0));
instructions.AddBitmap(KAKASHI612, RGB(255, 242, 0));
instructions.AddBitmap(KAKASHI613, RGB(255, 242, 0));
instructions.AddBitmap(KAKASHI614, RGB(255, 242, 0));
instructions.AddBitmap(KAKASHI615, RGB(255, 242, 0));
instructions.AddBitmap(KAKASHI616, RGB(255, 242, 0));
instructions.AddBitmap(KAKASHI617, RGB(255, 242, 0));
victory.AddBitmap(KAKASHI701, RGB(255, 242, 0));
victory.AddBitmap(KAKASHI702, RGB(255, 242, 0));
victory.AddBitmap(KAKASHI703, RGB(255, 242, 0));
victory.AddBitmap(KAKASHI704, RGB(255, 242, 0));
victory.AddBitmap(KAKASHI705, RGB(255, 242, 0));
victory.AddBitmap(KAKASHI706, RGB(255, 242, 0));
victory.AddBitmap(KAKASHI707, RGB(255, 242, 0));
victory.AddBitmap(KAKASHI708, RGB(255, 242, 0));
victory.AddBitmap(KAKASHI709, RGB(255, 242, 0));
victory.AddBitmap(KAKASHI710, RGB(255, 242, 0));
victory.AddBitmap(KAKASHI711, RGB(255, 242, 0));
victory.AddBitmap(KAKASHI712, RGB(255, 242, 0));
victory.AddBitmap(KAKASHI713, RGB(255, 242, 0));
victory.AddBitmap(KAKASHI801, RGB(255, 242, 0));
victory.AddBitmap(KAKASHI802, RGB(255, 242, 0));
victory.AddBitmap(KAKASHI803, RGB(255, 242, 0));
victory.AddBitmap(KAKASHI804, RGB(255, 242, 0));
victory.AddBitmap(KAKASHI805, RGB(255, 242, 0));
victory.AddBitmap(KAKASHI806, RGB(255, 242, 0));
victory.AddBitmap(KAKASHI807, RGB(255, 242, 0));
victory.AddBitmap(KAKASHI808, RGB(255, 242, 0));
victory.AddBitmap(KAKASHI809, RGB(255, 242, 0));
victory2.AddBitmap(vic_01, RGB(255, 242, 0));
victory2.AddBitmap(vic_02, RGB(255, 242, 0));
victory2.AddBitmap(vic_03, RGB(255, 242, 0));
victory2.AddBitmap(vic_04, RGB(255, 242, 0));
victory2.AddBitmap(vic_05, RGB(255, 242, 0));
victory2.AddBitmap(vic_06, RGB(255, 242, 0));
victory2.AddBitmap(vic_07, RGB(255, 242, 0));
victory2.AddBitmap(vic_08, RGB(255, 242, 0));
victory2.AddBitmap(vic_09, RGB(255, 242, 0));
victory2.AddBitmap(vic_10, RGB(255, 242, 0));
victory2.AddBitmap(vic_11, RGB(255, 242, 0));
victory2.AddBitmap(vic_12, RGB(255, 242, 0));
victory2.AddBitmap(vic_13, RGB(255, 242, 0));
victory2.AddBitmap(vic2_01, RGB(255, 242, 0));
victory2.AddBitmap(vic2_02, RGB(255, 242, 0));
victory2.AddBitmap(vic2_03, RGB(255, 242, 0));
victory2.AddBitmap(vic2_04, RGB(255, 242, 0));
victory2.AddBitmap(vic2_05, RGB(255, 242, 0));
victory2.AddBitmap(vic2_06, RGB(255, 242, 0));
victory2.AddBitmap(vic2_07, RGB(255, 242, 0));
```

```
		victory2.AddBitmap(vic2_08, RGB(255, 242, 0));
		victory2.AddBitmap(vic2_09, RGB(255, 242, 0));
		victory2.AddBitmap(vic2_10, RGB(255, 242, 0));
		victory2.AddBitmap(vic2_11, RGB(255, 242, 0));
		victory2.AddBitmap(vic2_12, RGB(255, 242, 0));
		victory2.AddBitmap(vic2_13, RGB(255, 242, 0));
		victory2.AddBitmap(vic2_14, RGB(255, 242, 0));
		victory2.AddBitmap(vic2_15, RGB(255, 242, 0));
		victory2.AddBitmap(vic2_16, RGB(255, 242, 0));
		victory2.AddBitmap(vic2_17, RGB(255, 242, 0));
		failed.AddBitmap(KAKASHI901, RGB(255, 242, 0));
		failed.AddBitmap(KAKASHI902, RGB(255, 242, 0));
		failed.AddBitmap(KAKASHI903, RGB(255, 242, 0));
		failed.AddBitmap(KAKASHI904, RGB(255, 242, 0));
		failed.AddBitmap(KAKASHI905, RGB(255, 242, 0));
		failed.AddBitmap(KAKASHI906, RGB(255, 242, 0));
		failed.AddBitmap(KAKASHI907, RGB(255, 242, 0));
		failed.AddBitmap(KAKASHI908, RGB(255, 242, 0));
		failed.AddBitmap(KAKASHI909, RGB(255, 242, 0));
		failed.AddBitmap(KAKASHI910, RGB(255, 242, 0));
		failed.AddBitmap(KAKASHI911, RGB(255, 242, 0));
		final.AddBitmap(final01, RGB(255, 242, 0));
		final.AddBitmap(final02, RGB(255, 242, 0));
		final.AddBitmap(final03, RGB(255, 242, 0));
		final.AddBitmap(final04, RGB(255, 242, 0));
		final.AddBitmap(final05, RGB(255, 242, 0));
		final.AddBitmap(final06, RGB(255, 242, 0));
		final.AddBitmap(final07, RGB(255, 242, 0));
		final.AddBitmap(final08, RGB(255, 242, 0));
		final.AddBitmap(final09, RGB(255, 242, 0));
		final.AddBitmap(final10, RGB(255, 242, 0));
	}
	bool Kakashi::GetEnd()
	{
		return isEnd;
	}
	bool Kakashi::IsFinish()
	{
		return isFinish;
	}
	bool Kakashi::IsDone()
	{
		return Done;
	}
}
```