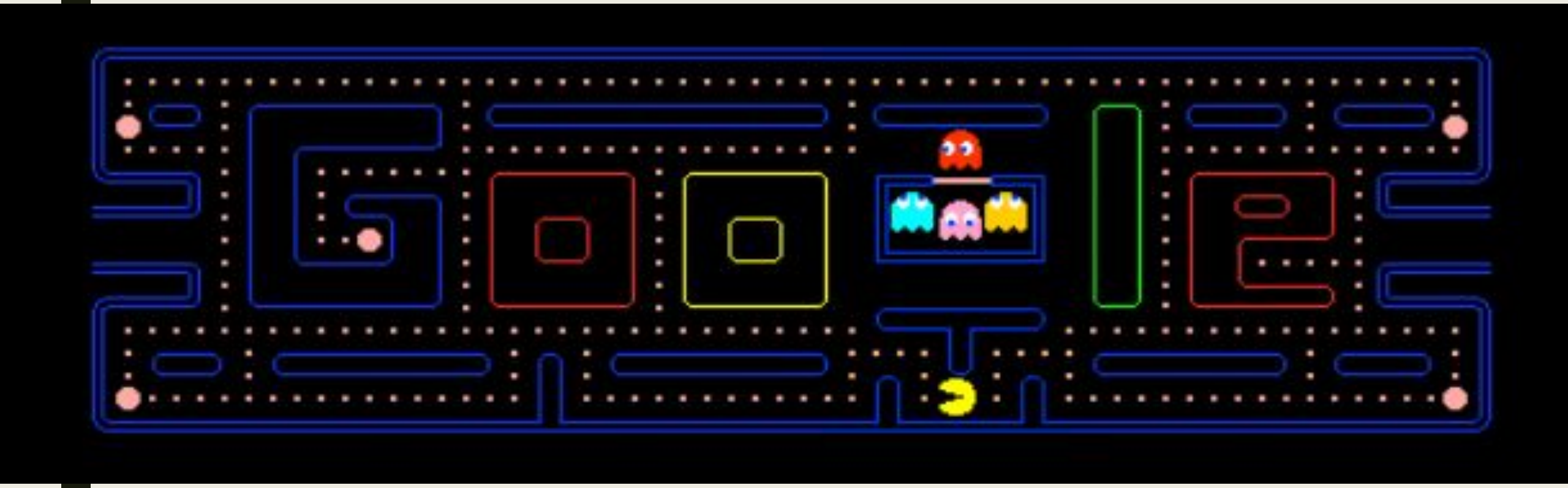
FINAL PROJECT RULES

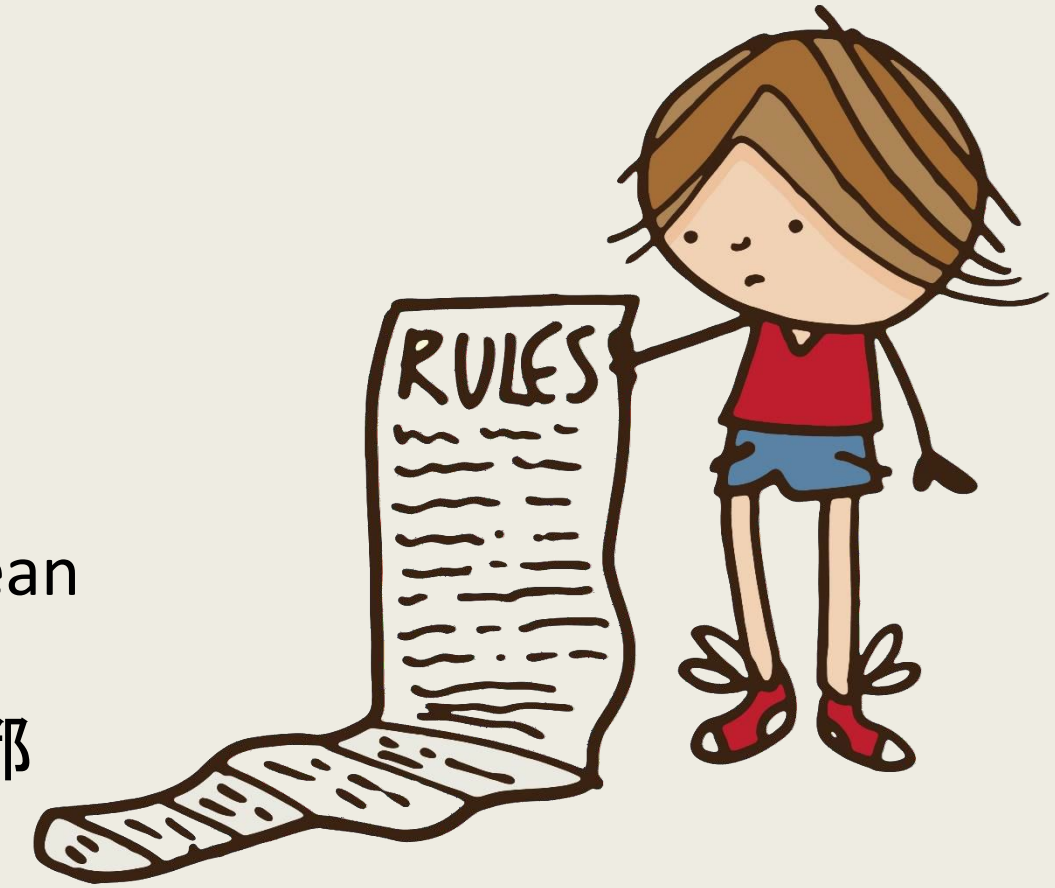
Introduction to Programming 2021


HI-SCORE
10000
440

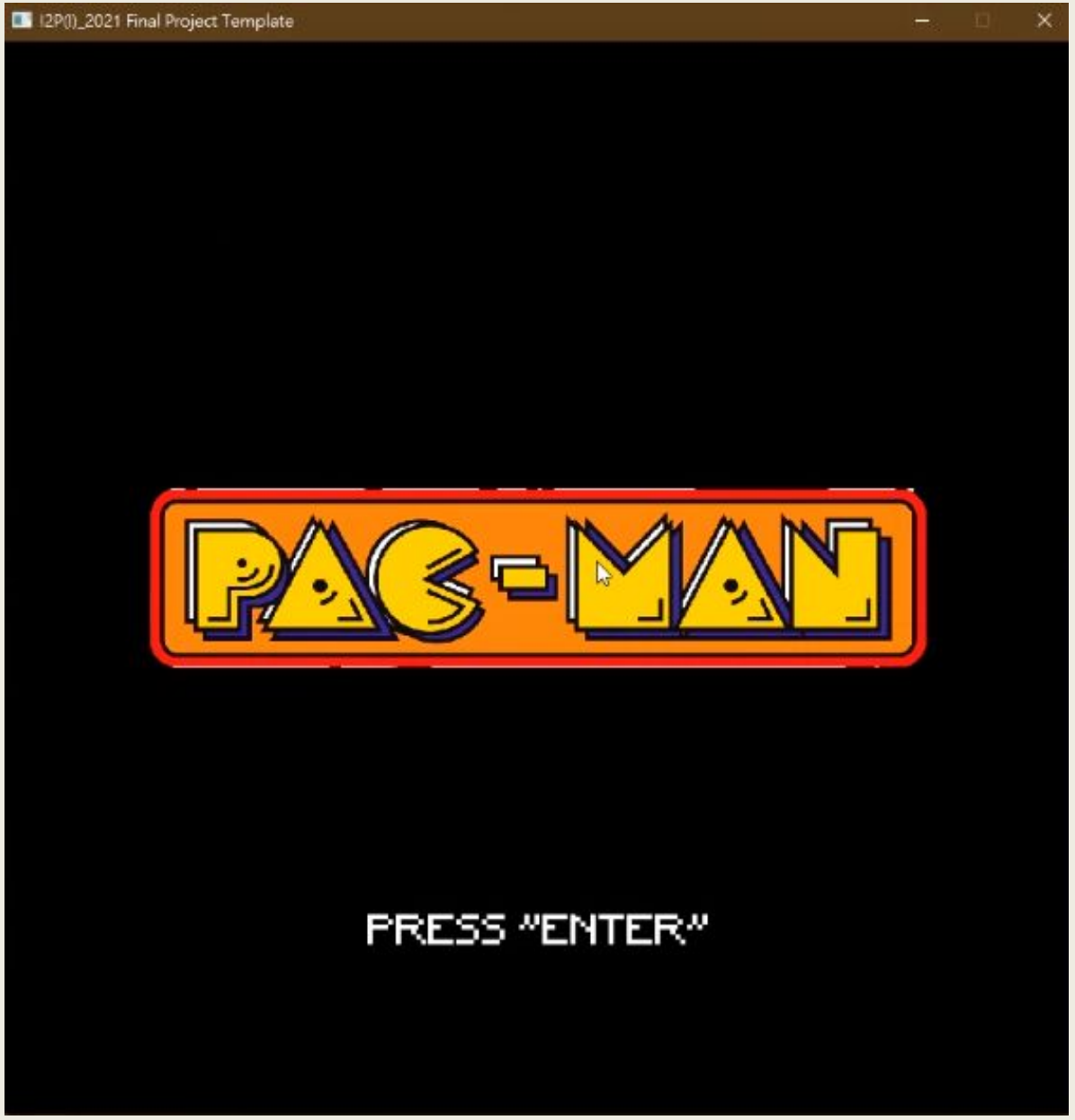


Rules

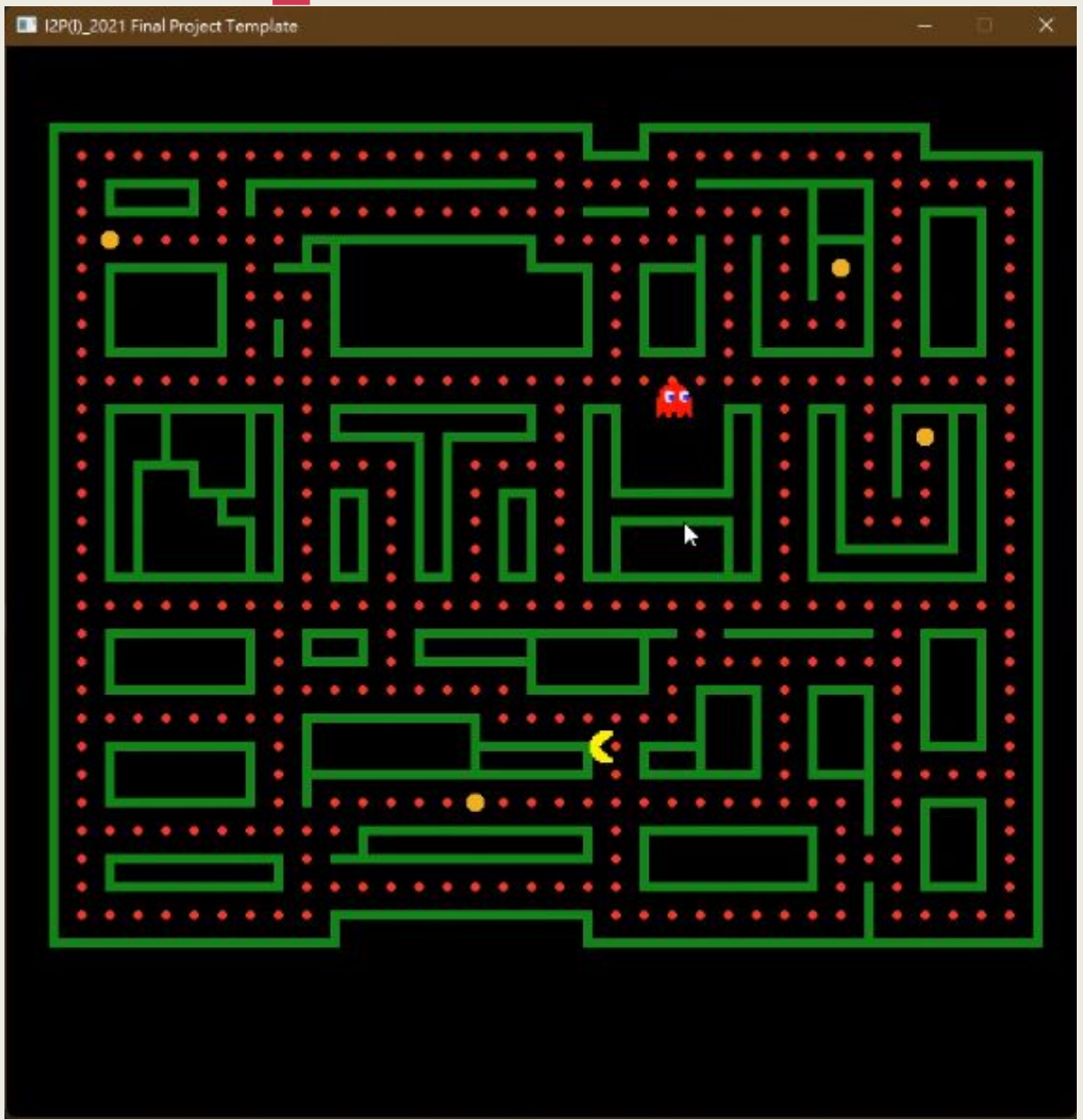
- 一人一組
- 占總成績20%
- 必須使用我們提供的template
- 2022年1月18、19日Demo
  - 需自備筆電
  - 詳細資訊前一周公布
- 用C語言，以及Allegro 提供 boolean value
  - C++, Python等其他程式語言都不行



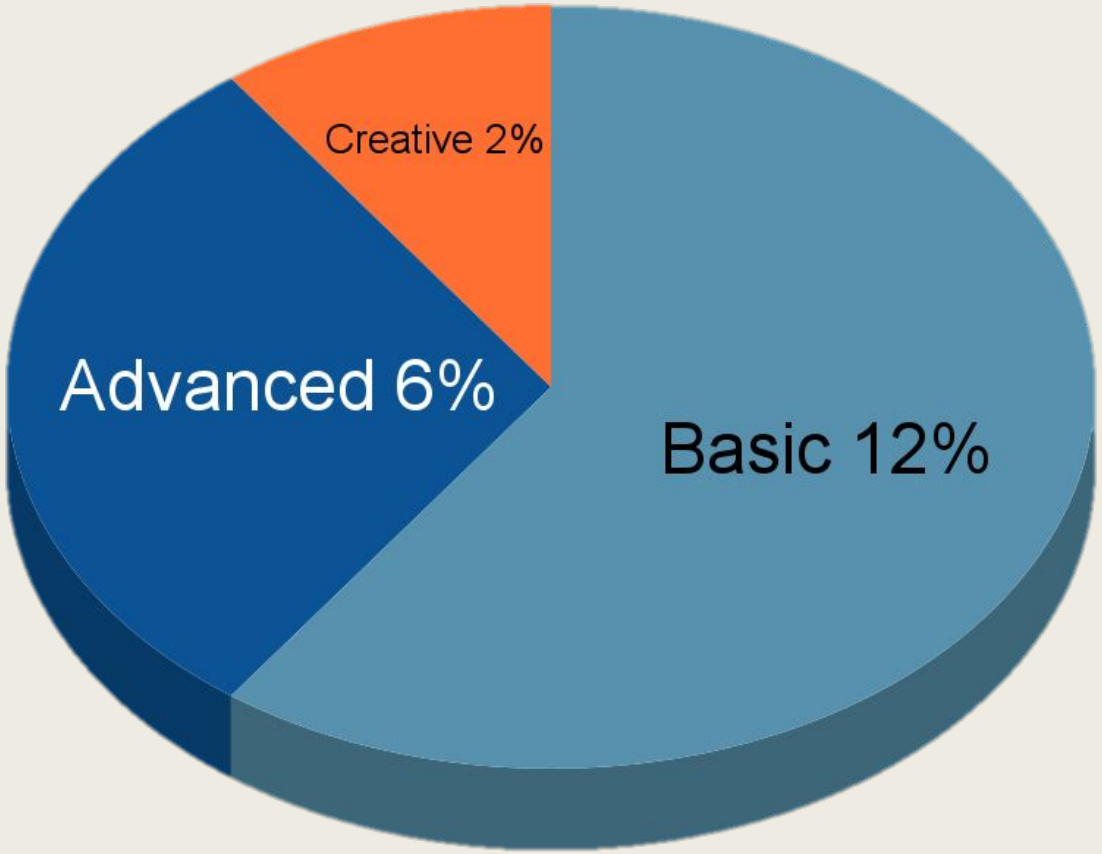
Given template



Given template



We’ ll finish 3% of basic features today



Basics (12%)

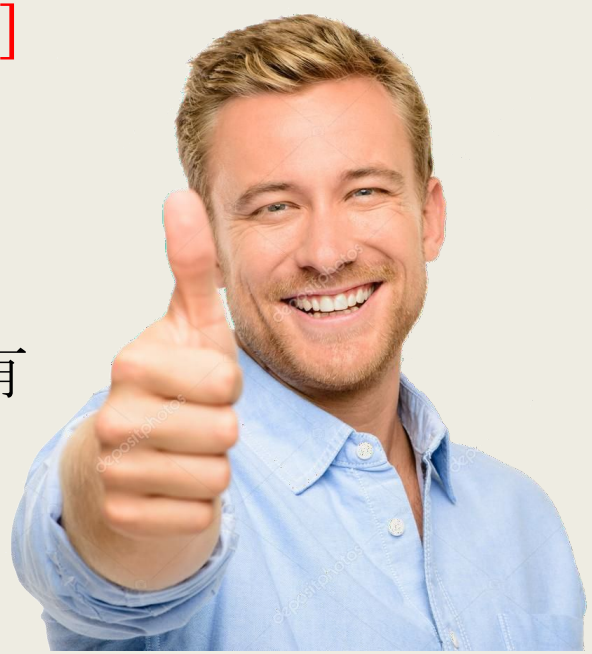
- 分為五個類別
- 遊戲完整性類別 3%
- 場景類別 2%
- 控制類別 2%
- 記憶體管理 2%
- HACKATHON 3%

Basics (12%) 遊戲完整性類別(3%)

- Pacman [HACKATHON-1]的正常移動 (不能出牆或是破圖)
- 基本吃豆子 [HACKATHON-1]
- Pacman碰到鬼要正常死亡, 吃完豆子要結束遊戲(或開始下一關)
- Ghost
  - Ghost的正常移動[HACKATHON-2] (不能出牆或是破圖)
  - Ghost 出場方式 [HACKATHON-2]
- 讀取.txt檔案生成map
- 遊戲中計分及顯示 吃豆子獲得分數
- Random移動的Ghost
  - 不能來回抖動
  - 不走回頭路

Basics (12%) 場景類別(2%)

- 三個原始場景Menu, Game, Setting [HACKATHON-3]
  - 要能正常轉場
  - Game結束後要能回到Menu或是轉往下個場景
    - 不可擅自關閉遊戲, 關閉遊戲的唯一條件只有滑鼠點擊關閉視窗or自行設計的EXIT UI
- 另外增加第四個場景
  - e.g Win, Game Over, Restart, End, etc.



Basics (12%)控制類別(2%)&記憶體管理(2%)

- 使用滑鼠(ex.點擊進入Setting場景)[HACKATHON-3]以及鍵盤控制pacman移動[HACKATHON-1]
  - Setting 要有音量、音效調節
- 記憶體管理
  - Memory 使用量的最大值是固定的
  - 即記憶體回收管理, 不可無上限的增加記憶體用量
  - (實際demo的測試方式, 助教會透過profiler觀看記憶體用量)

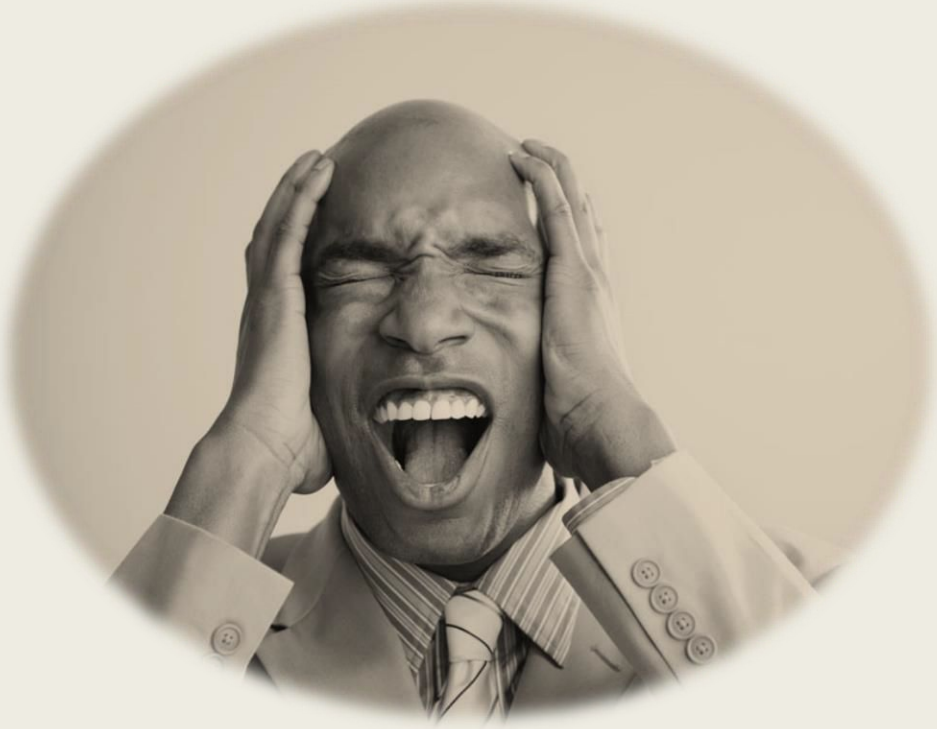
Basics (12%) HACKATHON(3%)

- (Those marked in red in previous slides)
- HACKATHON 1
  - Pacman 的正常移動以及吃豆子
- HACKATHON 2
  - Ghost 的出場和 random 移動 (會走回頭路)
- HACKATHON 3
  - 利用滑鼠點擊進入Setting頁面

Advance (6%)

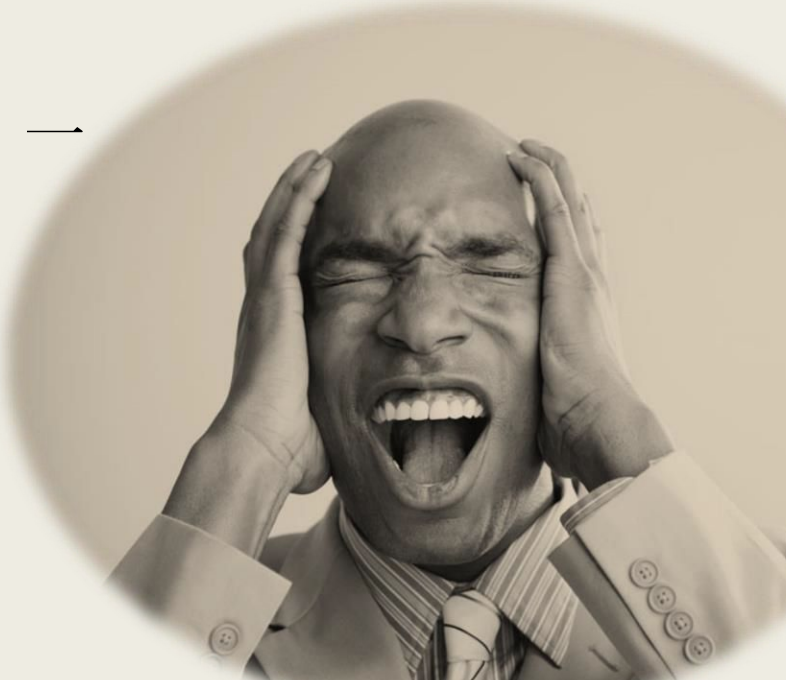
- Power Bean 大力丸 (2%)
- 設計其他Ghost的追擊策略(1%)
- 角色動畫(ex. pacman的開合開合, Ghost的移動動畫) (1%)
- 美術類別 (1%)
- 遊戲性類別 (1%)
- 功能、介面性類別(1%)

(註: 這邊最多就是拿滿6%)



Advance (6%)Power Bean (2%)

- 實作Classic Pacman的大力丸機制
  - 吃到之後進入一定時間的 可以反過來吃掉鬼魂的力量。鬼魂在這段時間內會變成深藍色(有提供sprite) 速度變慢
  - 大力丸的效果時間快到的時候鬼應該要藍白閃爍
    - 此時鬼魂的策略應是遠離小精靈
  - 鬼魂被吃到要跑回去籠子, 跑的過程要變眼睛圖示, 一段時間後出來
  - 可參考google pacman
  - 有關Ghost的詳細State Machine在下一張Slide



Advance (6%)Power Bean (2%)

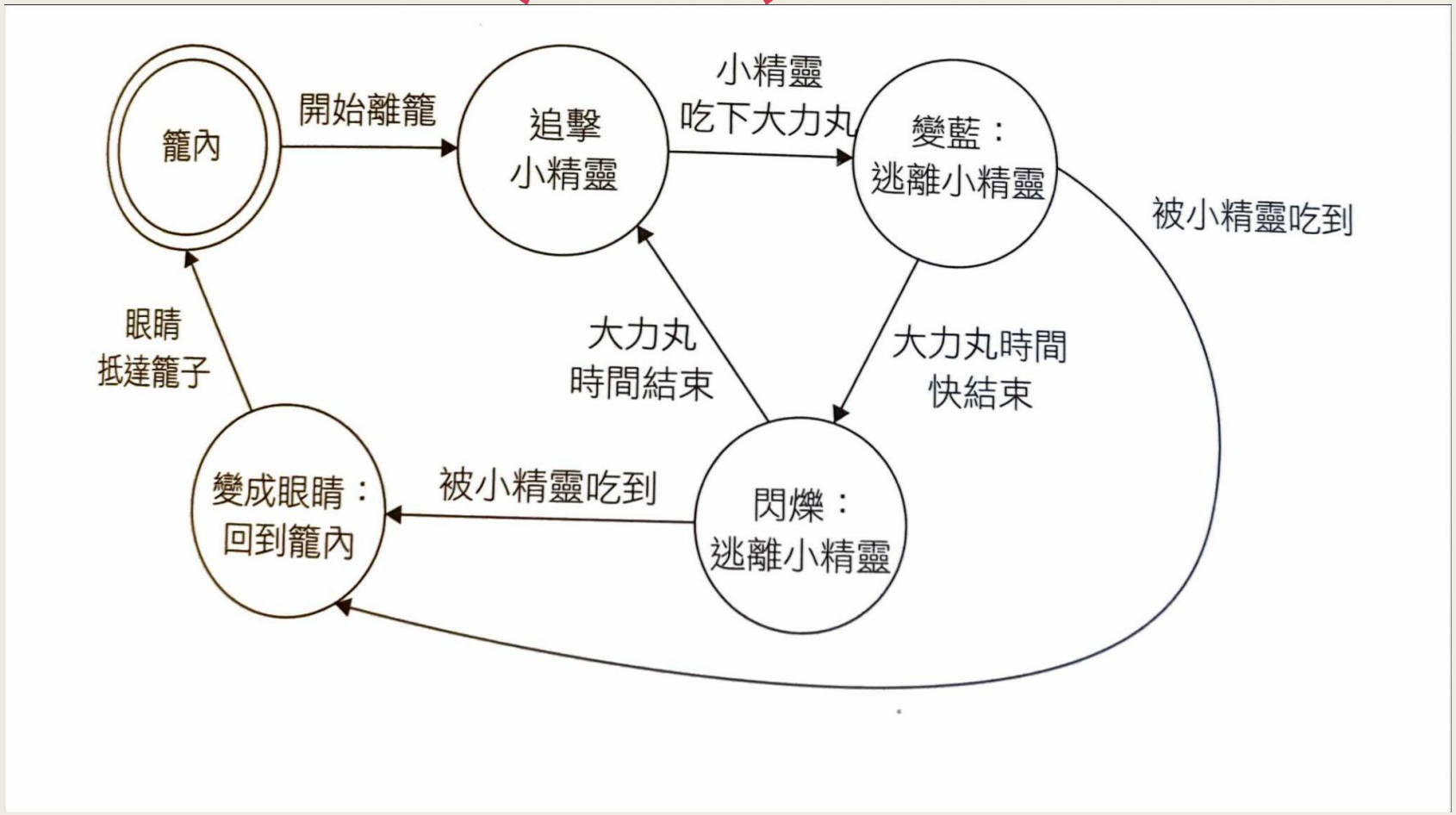
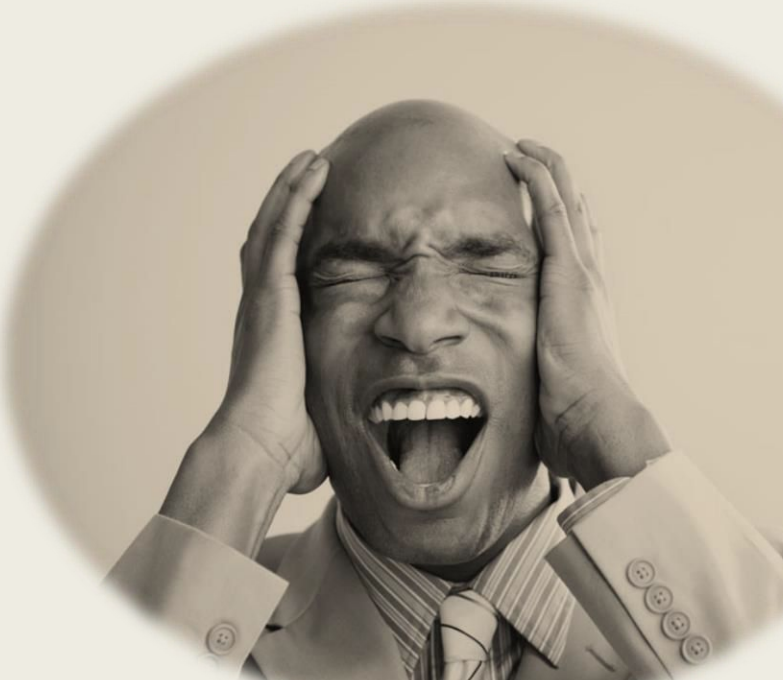


Image Reference

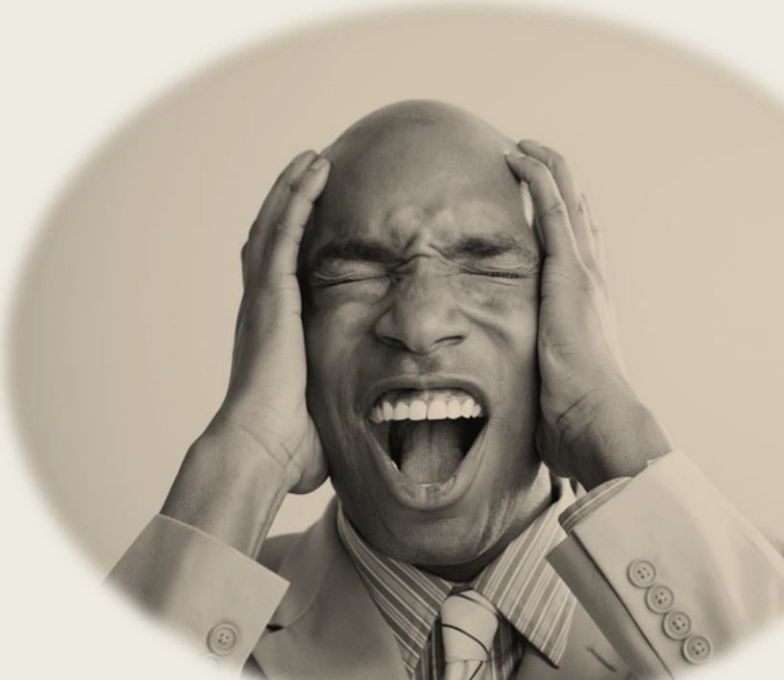
Advance (6%) 角色動畫(1%)

- pacman的開合動畫
- Ghost的移動動畫
- 可利用的sprites都已經放在 Assets裡面了
- 也可以自行使用其他的sprites
  - 請確保至少有四個方向, 各個方向兩個以上(包含兩個)的sprites



Advance (6%) 美術類別(1%)

- 不同場景有不同的BGM
- 不同版本的音效
- 美化 UI
- 美化地圖
- 其他任何美術相關
- (以上選兩項即可)
  - (其他美術相關請自行列舉)



Advance (6%) 遊戲性類別(1%)

- 自行設計另外兩個道具並設計相關功能
- 可以接受的功能範例
  - 吃到道具後，提升移動速度
  - 吃到道具後，暫時獲得穿牆能力
  - 吃到道具後，開啟傳送門
- 不能接受的範例
  - 吃到道具後+50分
  - 吃到道具+秒數

註* 藍綠二擇一即可

- 可選擇地圖or多關卡設計
- 多人遊戲(2P以上合作破關)

Advance (6%) 功能、介面性類別(1%)

- 遊戲內角色選擇
- 永久計分並設計排行榜

註* 兩項都須達到

Creative (2%)

- **角色精細度**
- **技能華麗度**
- **動畫炫泡度**
- **遊戲豐富度**
- **整體流暢度**
- 上述以及Advance以外，任何你覺得超酷或超有 Implement難度的功能都可以實作出來並在Demo時候解釋給助教做評分

Template

- Multiple file template
  - *Template .zip*
  - functions and scenes are separated to different files.

Template (if you use Allegro 5.0)

- Change

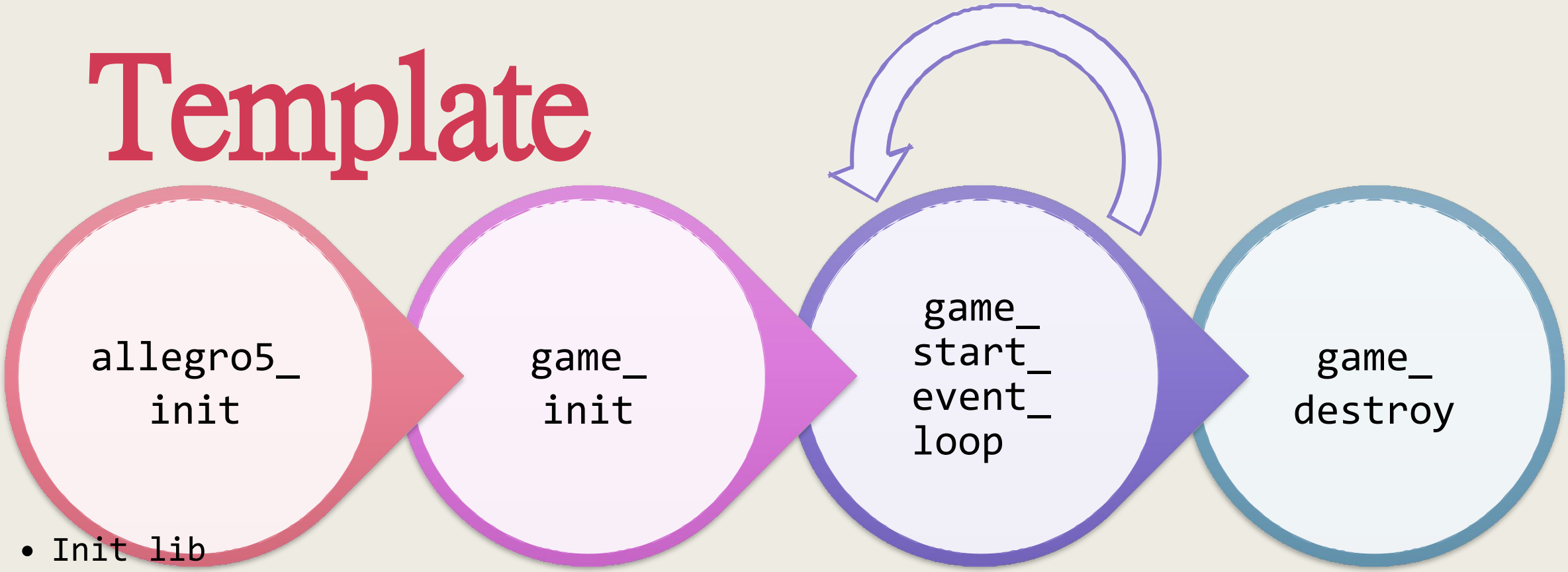
```
if (!al_init_font_addon())
    game_abort("failed to initialize font add-on");
```

to

```
al_init_font_addon();
```

- (You only need to fix this if you followed the tutorial that uses **allegro-5.0.10-monolith-mt.dll**)

Template



- Init lib routines
- init/install
- create display, event queue, timer
- register events
- start timer

- Init variables
- load resources
- change scene
  - to main scene

- Process events
- close window
- timer
  - update
  - draw
- keyboard events
- mouse events

- Free variables
- free resources
- change scene to main scene

Template(states)

```
// The active scene id.
int active_scene;
// Keyboard state, whether the key is down or not.
bool key_state[ALLEGRO_KEY_MAX];
// Mouse state, whether the key is down or not.
// 1 is for left, 2 is for right, 3 is for middle.
bool *mouse_state;
// Mouse position.
int mouse_x, mouse_y;
```

Template(structs)

```
typedef struct object {
    Pair_IntInt Coord; //
    Pair_IntInt Size; // x for width, y for height
    Directions facing;
    Directions preMove;
    Directions nextTryMove;
    uint32_t moveCD;        // movement CountDown
} object;
```

Template(enum)

```
typedef enum Directions{
    NONE = 0,     UP = 1,
    LEFT = 2,     RIGHT = 3,
    DOWN = 4,     UP_DOWN = 5,
    LEFT_RIGHT = 6,     UP_LEFT = 7,
    DOWN_LEFT = 8,     DOWN_RIGHT = 9,
    UP_RIGHT = 10
} Directions;
```

Template(struct)

```
typedef struct RecArea{
    float x, y, w, h;
} RecArea;
typedef struct Pair_IntInt {
    int x;
    int y;
} Pair_IntInt;
```

```
typedef struct bitmapdata{
    int bitmap_x;
    int bitmap_y;
    int bitmap_w;
    int bitmap_h;
} bitmapdata;
```

Template(structs)

```
typedef struct Pacman{
    bitmapdata imgdata;
    object objData;
    func_ptr move;
    int speed;
    bool powerUp;
    ALLEGRO_TIMER* death_anim_counter;
    ALLEGRO_BITMAP* move_sprite;
    ALLEGRO_BITMAP* die_sprite;
} Pacman;
```

Template(routines)

```
// Initialize allegro5 library
void allegro5_init(void);
// Initialize variables and resources.
void game_init(void);
// Process events inside the event queue using an infinity loop.
 void game_start_event_loop(void);
// Release resources.
void game_destroy(void);
// Function to change from one scene to another.
void game_change_scene(int next_scene);
```

Template(events/callbacks)

```
// This is called when the game should update its logic.
void game_update(void);
// This is called when the game should draw itself.
void game_draw(void);
void on_key_down(int keycode);
void on_mouse_down(int btn, int x, int y);
```

Template (utilities/callbacks)

```
// Load resized bitmap and check if failed.
ALLEGRO_BITMAP *load_bitmap_resized(const char *filename, int w, int h);
// Display error message and exit the program, used like 'printf'.
// Write formatted output to stdout and file from the format string.
// If the program crashes unexpectedly, you can inspect "log.txt" for
// further information.
void game_abort(const char* format, ...);
// Log events for later debugging, used like 'printf'.
// Write formatted output to stdout and file from the format string.
// You can inspect "log.txt" for logs in the last run.
void game_log(const char* format, ...);
```

Template (draw)

```
RecArea drawArea = getDrawArea(pman->objData, GAME_TICK_CD);

    //Draw default image
    al_draw_scaled_bitmap(pman->move_sprite, 0, 0,
        16, 16,
        drawArea.x + fix_draw_pixel_offset_x,
    drawArea.y + fix_draw_pixel_offset_y,
        draw_region, draw_region, 0
    );
```

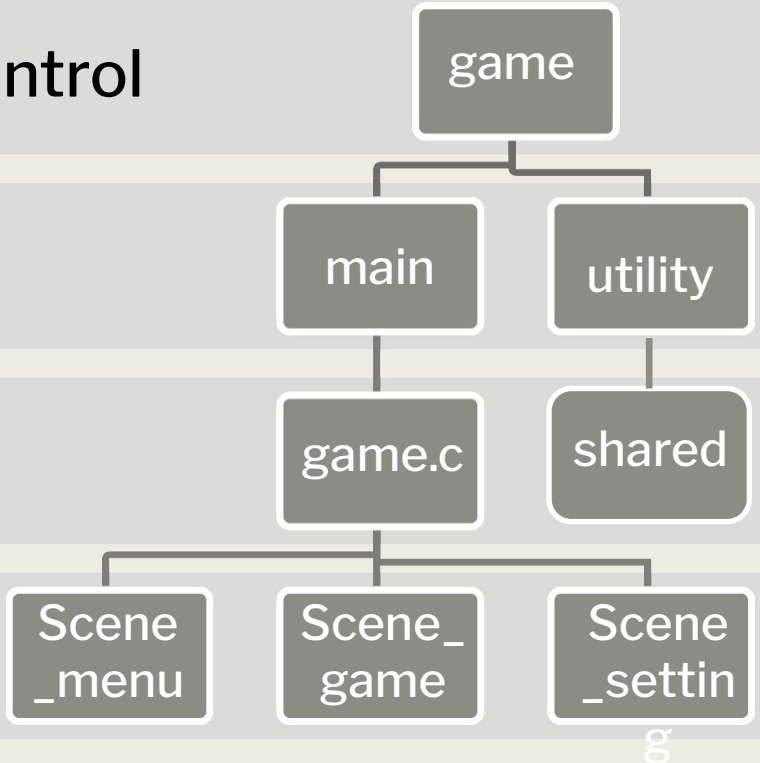
Template Structure

Allegro5 routines & Scene control

Utility functions & Entry point

Shared resources

Scenes



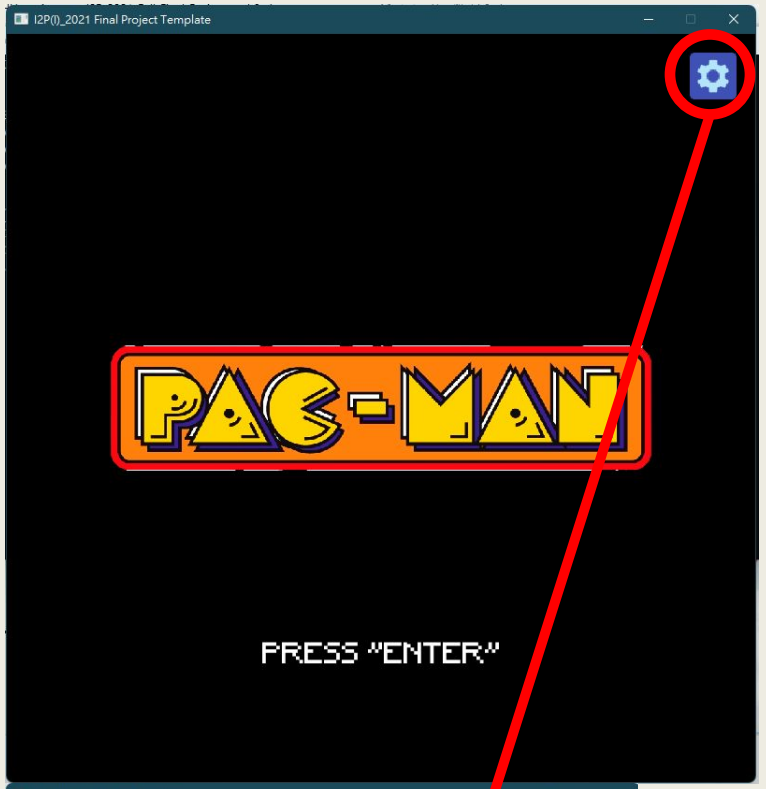
Today’s Goal

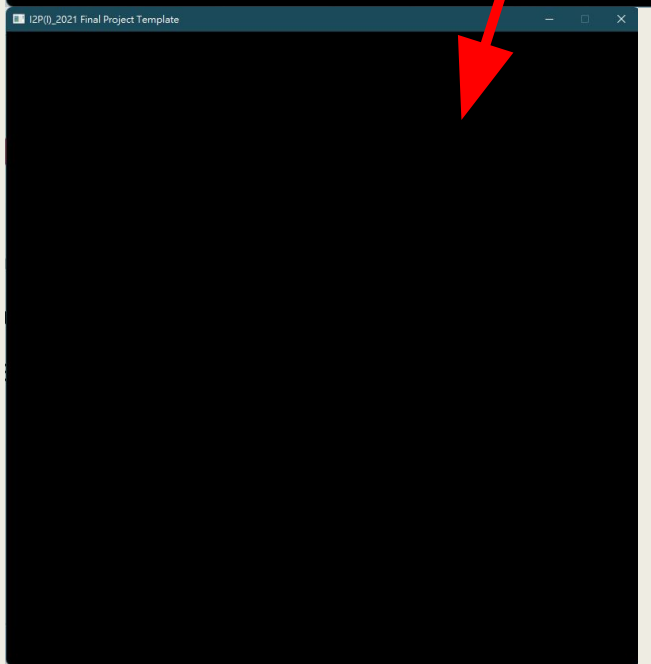
- Pacman Movement and Eat Bean
- Ghost Go Out & random movement (may go back and forth)
- Mouse event(Click) and enter setting scene

  - *Create the settings scene.*
    (can be entirely black with no functions)
  - *A button in main scene. (w/ mouse in/out animation)*

Today’ s Goal

- Pacman Movement and Eat Bean
- Ghost Go Out & random movement (may go back and forth)
- Mouse event(Click) and enter setting scene
  - *Create the settings scene.*
    (can be entirely black with no functions)
  - *A button in main scene.*
    (w/ mouse in/out
    animation)





Today’s Goal (Example)

- For today’s goal, you only need to uncomment the codes and replace the “???” with the correct code.

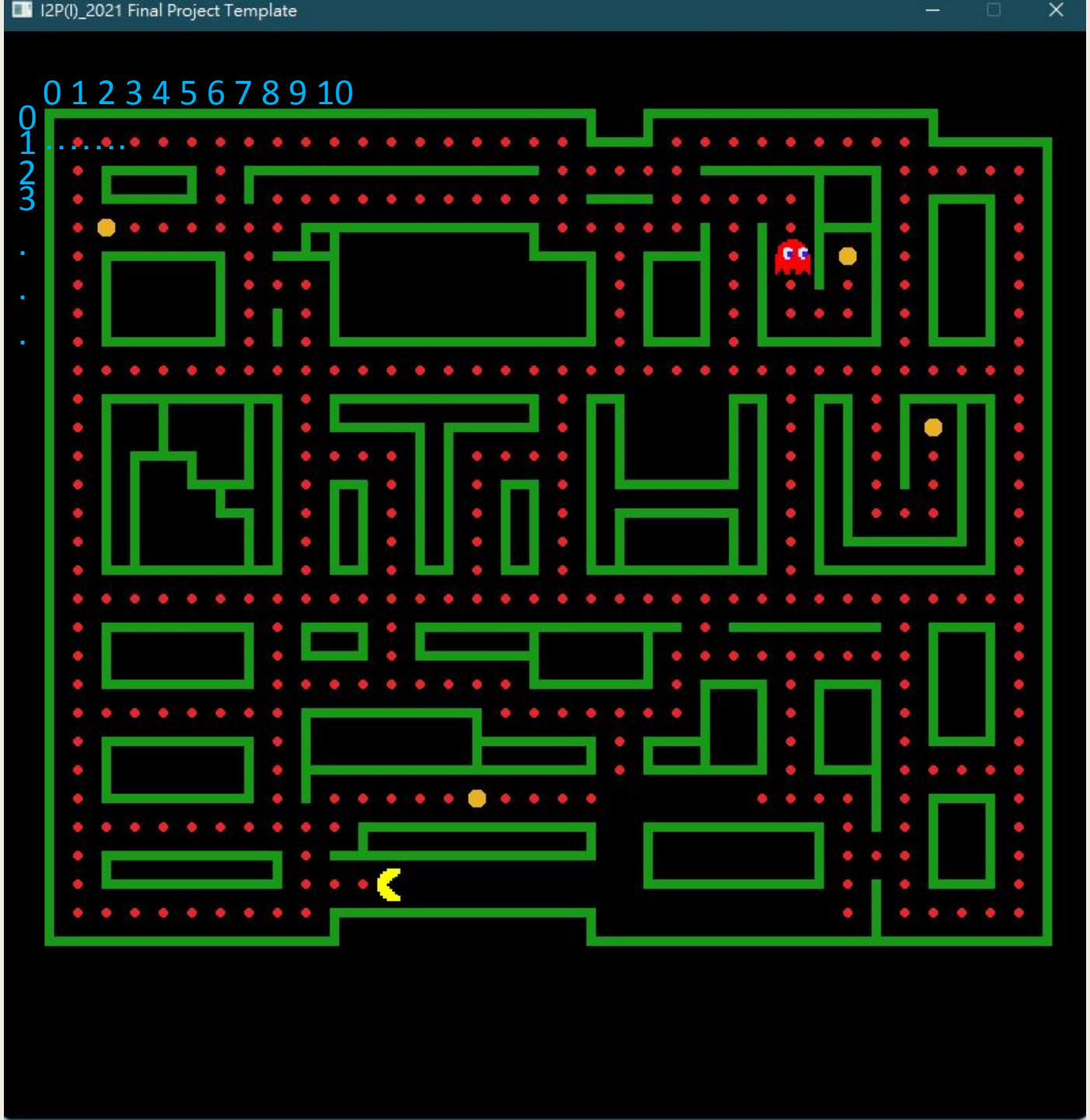
```
// [HACKATHON 1-1]
        // TODO: Use allegro pre-defined enum ALLEGRO_KEY_<KEYNAME> to controll
pacman movement
        // we provided you a function `pacman_NextMove` to set the pacman's next
move direction.
        /*
        case ALLEGRO_KEY_W:
            pacman_NextMove(pman, ...);
            break;
        case ALLEGRO_KEY_A:
            pacman_NextMove(pman, ...);
            break;
        case ALLEGRO_KEY_S:
            pacman_NextMove(pman, ...);
            break;
      .........
```

Today’s Goal (I)

- Setup movement for your pacman
- (HACKATHON 0-1) line 161 in map.c for loading map
- [HACKATHON] 1-1 ~ 1-4
- Separate the x and y axes. Use the same calculation to detect each axis.

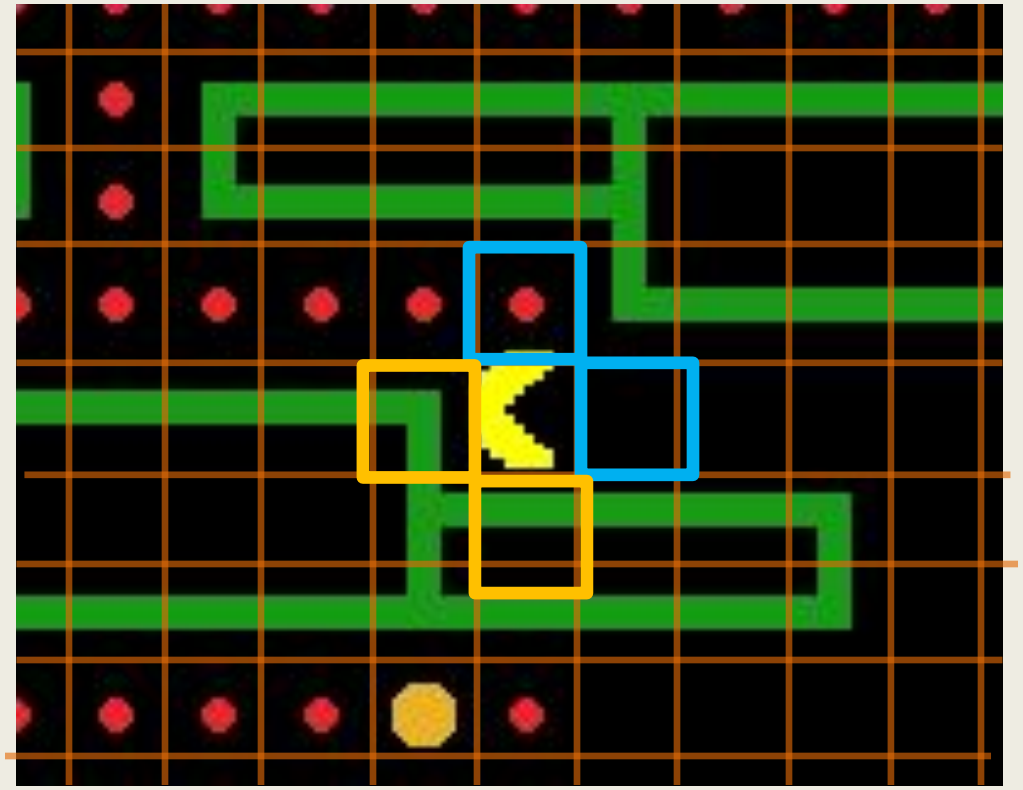
```
// [HACKATHON 0-1]
            // You can just switch to nthu_map if you want to finish HACKATHON 0 later.
                M->map[i][j] = default_map[i][j];
            //  M->map[i][j] = nthu_map[i][j];
```

Today's Goal (I)



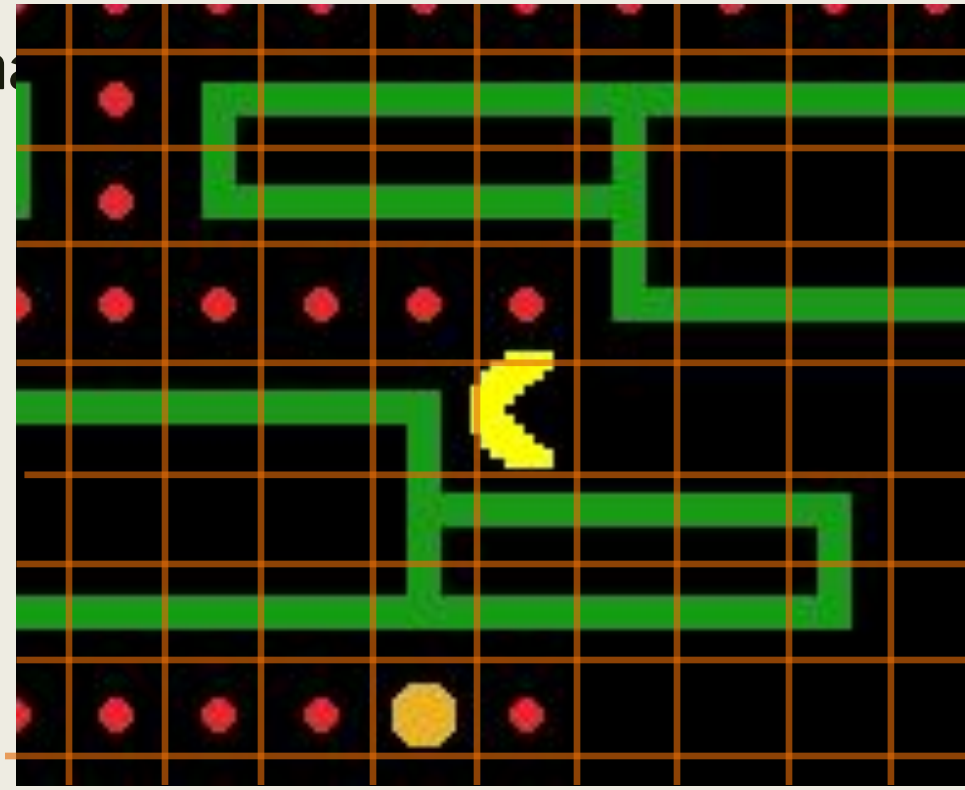
Today’s Goal (I)

- [HACKATHON 1-2] Setup Check of valid movement in pacman_movable(...)
- Valid
- Non-Valid



Today’ s Goal (I)

- [HACKATHON 1-3~4] Use `pacman_eatItem(...)` to activate the effect of item. (Playing sound)
- And erase the item from 2-D char Array ma



Today’s Goal (II)

- Allocate ghosts. (Today, one ghost is enough.
- Let Ghost start to move.
- [HACKATHON] 2-0 ~ 2-4
- Control the state of ghost
- `ghost_movable` use the same logic of `pacman_movable`
- Today, only focus on the `ghost_red_move_script_FREEDOM` function.
  - But the state machine of ghost movement is important for your future programming.

```
typedef enum {
    BLOCKED,
    GO_OUT,
    FREEDOM,
    GO_IN,
    FLEE
} GhostStatus;
```

Today’s Goal (III)

- Implement a new scene
  - *Create the settings scene. (can be entirely black with no functions)*
  - *A button in main scene. (with mouse in/out animation)*
- [HACKATHON] 3-1 ~ 3-9

In game_change_scene, game_update, game_draw, on_key_down, …

```
if (active_scene == SCENE_MENU) {
    //…
} else if (active_scene == SCENE_START) {
    //…
} else if (active_scene == SCENE_SETTINGS) {
```

Today’ s Goal

- Aside from filling the blanks, make sure you understand the entire game flow and how each code section works.
- Find a TA and demo the 3 goals to get 3% score.
- The TA will ask you to explain how the 3 goals are implemented, you’ll get 3% score if you can describe how the code works.
- (each goal deserves 1% score respectively)

Useful Resource

- **Allegro 5 Wiki**
  - https://www.allegro.cc/manual/5/
- **Allegro 5 reference manual**
  - https://liballeg.org/a5docs/trunk/
- **Movie Tutorial**
  - https://www.youtube.com/watch?v=IZ2krJ8Ls2A&list=PL6B459AAE1642C8B4
- **2D Game Development Course**
  - http://fixbyproximity.com/2d-game-development-course/

DON’ T CHEAT

LET’S CODE

Have a nice day~