

# Hangman小遊戲

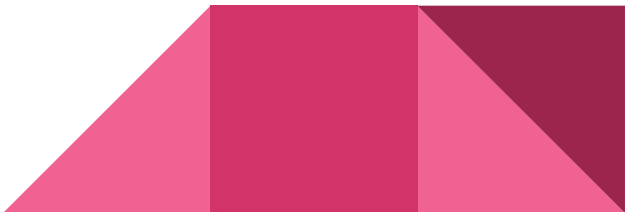
組員：林宸宇、蕭凡凱、林彥安、許家頂、高芙琪  
遊戲製作：林宸宇  
簡報製作：林宸宇

# 研究動機

我們想在生活中活用演算法，運用老師教授的思考模式解決問題，於是決定將所學應用在設計遊戲。




# 研究目的

1. 希望藉由設計遊戲，使我們領略在處理問題時思考步驟的重要性，並同時訓練到大腦的邏輯思維。
  2. 熟悉掌握各個演算法，加以連結運用，讓演算法實際應用在生活當中。
  3. 訓練我們在面對問題時，要先拆解問題，並逐一破解。
- 

## Hangman先備知識

Hangman是一款猜英文單字的雙人遊戲。出題者會出一道題，題目為一個英文單字。出題者在遊戲開始時應畫上單字字母數量個底線，猜題者則要提出單一字母詢問。若猜題者答對，出題者應在所有出現該字母的底線上標出字母。若猜題者答錯，則出題者應為Hangman添上一個部位。若Hangman全身所有部位都出現(死亡)，則遊戲結束，猜題者失敗。若Hangman尚死亡且整個單字已被猜出，則猜題者獲勝。

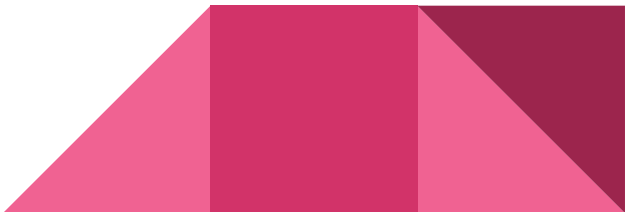


# 研究方法

Hangman這款遊戲需要猜題者的多筆詢問以及許多判斷。我們先把複雜的流程簡化成小問題，再逐一處理。



## 研究方法--拆解問題

1. 如何讓出題者出題，又不讓使用者看到題目？
  2. 如何產生底線讓使用者填入字母？
  3. 如何判斷使用者輸入的字母是否存在字中，若存在，則如何將所有出現位置代換成該字母？
  4. 如何將hangman目前的身體用電腦輸出？
- 

這些問題讓我們來  
一一破解！

# 研究方法--解決問題1

讓輸入的英文字母與猜題區之間的間隔拉開，因此需執行很多次的換行來把題目刷掉





## 研究方法--解決問題2

1. 記錄題目字的長度(以下稱len)
2. 造出一個長為len的陣列(以下稱lines), 將每一項放入底線  
ex: len = 5, lines = \_ \_ \_ \_ \_



## 研究方法--解決問題3

1. 共有6次猜錯的機會，只要還有機會就在每一輪讓使用者輸入字母



## 研究方法--解決問題3

1. 共有6次猜錯的機會，只要還有機會就在每一輪讓使用者輸入字母
2. 掃過題目，看到同個字母的時候把lines中同位置的底線換成字母



## 研究方法--解決問題3

1. 共有6次猜錯的機會，只要還有機會就在每一輪讓使用者輸入字母
2. 掃過題目，看到同個字母的時候把lines中同位置的底線換成字母
3. 若2.沒找到，則扣一次機會且多顯示一個hangman的部位  
若2.有找到，則顯示更新後的lines陣列，並判斷lines中是否還有底線，若以沒有底線則判斷為遊戲結束：獲勝



## 研究方法--解決問題4

用字元搭配換行將其拼湊成人的形狀

ex:

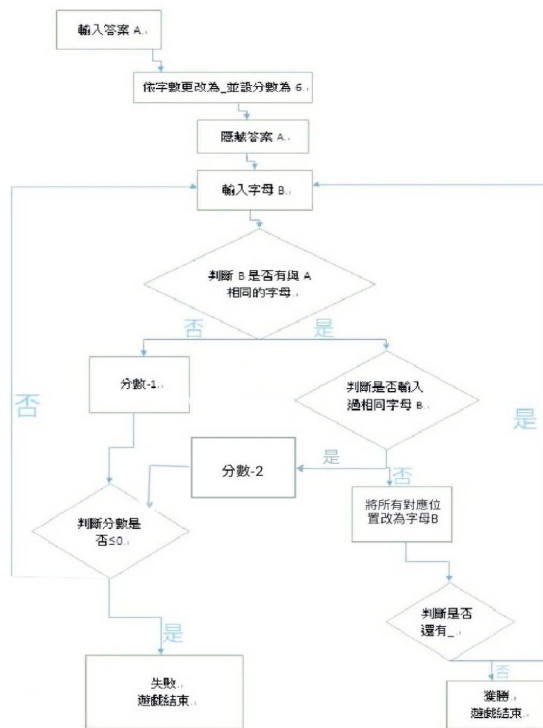
```
o
-|-
/\
```

Hi I'm hangman yay



好耶我們把問題都破光了  
來畫流程圖吧

# 流程圖



orz  
orz  
orz  
orz  
orz  
orz  
orz  
orz  
orz  
orz

流程圖畫完了  
來寫程式囉！  
成品產出-----



## 研究結果

根據前面所發想的輸入英文字母與猜題區之間的間隔拉開，需執行很多次的換行，程式碼如下：

```
printf("\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n");
```



# 研究結果

需記錄題目字的長度，程式碼如下：

```
unsigned long length = word.size();
```



## 研究結果

需用字元搭配換行將hangman 的身體用電腦輸出，程式碼如下：

```
const string manBodyParts[6] = {" o", "\n-", " |", " -", "\n /", " \\"};
```



## 研究結果

需造出一個長為len的陣列(以下稱lines)，將每一項放入底線，程式碼如下：

```
lines = vector<char>(length, '_');
```



# 研究結果

為了讓猜題者知道遊戲規則，程式碼如下：

```
printf("The word has %lu letters.\nFill in the blanks (one letter at a  
time for better experiences) before the whole Hangman appears on  
the screen (6 chances: 1 head, 2 arms, 1 body, 2 legs):\n", length);
```



## 研究結果

共有6次猜錯的機會，只要還有機會就在每一輪讓使用者輸入字母，程式碼如下：

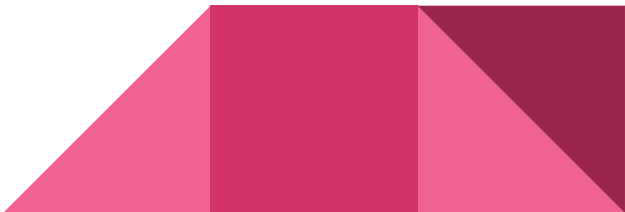
```
while (chances > 0) {  
    (void)!scanf("%c", &letter);
```



# 研究結果

如果題目中有該字母，則把 `lines` 中同位置的底線換成字母，並判斷是否還有底線，若已沒有底線則判斷為遊戲結束：獲勝。程式碼如下：

```
if (found) {  
    for (int i : idxs) lines[i] = letter;  
    printf("Correct! Current word:\n");  
    bool win = true;  
    for (char c : lines) {  
        if (c == '_') {  
            win = false;  
            break;  
        }  
    }  
    if (win) break;  
}
```



## 研究結果

若2.沒找到，則扣一次機會且多顯示一個hangman的部位，程式碼如下：

```
else {  
    chances--;  
    printf("Wrong letter! Hangman:\n");  
    printMan();
```





## 研究結果

檢查結果是勝是敗(是否還有剩餘機會):

```
if (chances <= 0) printf("You lose! Hangman is dead because of  
you! It's all your fault!\n");  
else printf("You win! You saved Hangman! Did you get lucky or are  
you too smart?\n");
```



程式寫完了  
來看看有沒有蟲蟲  
debug時間到

## 自我省思--發現問題

我們反覆測試程式後發現一個問題：如果挑戰者重複輸入同一個正確的字，那麼遊戲將無法結束，落入無限循環中。ex：輸入 e 系統判定正確，再次輸入 e 時，系統依然判定正確，結果也沒有改變，且猜錯機會也沒有減少。



## 自我省思--解決問題

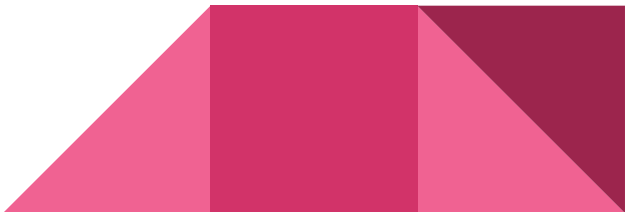
為了解決這一個問題，我們將輸入過的每一個字母儲存，並與接下來輸入的字母對照，如果有重複的字母，將會被視為作弊，猜錯的機會扣掉兩次，成功地解決了這一個問題。



# 自我省思--解決問題

附上程式碼：

```
bool cheat = false;
for (char c : typedLetters) {
    if (letter == c) {
        printf("You've tried this letter already. Are you trying to break this program or you literally forgot which letters  
you've already tried?\nAs a punishment, you lose 2 chances. Have fun hacking... LOL\n");
        chances -= 2;
        printMan();
        cheat = true;
        break;
    }
}
if (cheat) continue;
typedLetters.push_back(letter);
```



## 自我省思--反思、心得

經過這次的研究後，我們學到了解決問題並實作需要具備的能力：拆解問題的技巧、程式設計，但這些都需要人類的邏輯思考才能實現。我們的成品中就包含了許多思維，其中的演算法即展現了我們邏輯思考的過程，包含了我們所有組員的巧思。實作部分也讓我們對程式設計的認知進步了不少，在思考之後實際的執行確定了我們演算法的可行性。只要善用演算法邏輯思考，任何複雜問題都能被解決。



## 自我省思--反思、心得縮減版

只要善用演算法邏輯思考，任何複雜問題都能被解決。

這句話也是整學期上課的核心啊！



# 完整程式碼(太長了也只好.....先擠再說吧)

```
#include <bits/stdc++.h>
using namespace std;

char rawword[100];
string word;
vector<char> lines;
const string multiBodyPart[6] = {" a", "a ", "a ", " ", "a ", "a "};
int chances = 6;

void print_line() {
    for (char c : lines) {
        printf("%c", c);
    }
    printf("\n");
}

void printMan() {
    for (int i = 0; i < 6 - chances; i++) {
        printf("%s", multiBodyPart[i].data());
    }
    printf("a");
}

int main() {
    (void)scanf("%s", rawword);
    word = rawword;

    printf("a a a a a a a a a a a a a a a a");

    unsigned long length = word.size();
    lines = vector<char>(length, '_');

    char letter;
    printf("The word has %lu letters, %d fill in the blanks (one letter at a time for better experiences) before the whole Hangman appears on the screen (6 chances: 1 head, 2 arms, 1 body, 2 legs)%n", length);

    vector<char> typed_letters;
    while (chances > 0) {
        (void)scanf("%c", &letter);
        if (letter == '\n') continue;
        bool cheat = false;
        for (char c : typed_letters) {
            if (letter == c) {
                printf("You've tried this letter already. Are you trying to break this program or you literally forgot which letters you've already tried? %s As a punishment, you lose 2 chances. Have fun hacking... LOL %s",
                    chances == 2 ?
                    "punishment":
                    "cheat = true";
                break;
            }
        }
        if (cheat) continue;
        typed_letters.push_back(letter);
        bool found = false;
        vector<int> idxs;
        for (int i = 0; i < length; i++) {
            if (letter == word[i]) {
                found = true;
                idxs.push_back(i);
            }
        }
        if (found) {
            for (int i : idxs) lines[i] = letter;
            print_line();

            bool win = true;
            for (int c : lines) {
                if (c == '_') {
                    win = false;
                    break;
                }
            }
            if (!win) break;
        } else {
            chances--;
            printf("Wrong letter! Hangman %s",
                printMan());
        }
    }
    if (chances == 0) printf("You lose! Hangman is dead because of you! It's all your fault %s",
        the printf("You win! You saved Hangman! Did you get lucky or are you too smart? %s",
```



## 參考文獻

<https://zh.wikipedia.org/wiki/猜單詞遊戲>  
(Hangman的完整規則)



The background is a solid pink color. In the top right corner, there is a decorative pattern of overlapping triangles in various shades of pink and magenta, creating a geometric, abstract design.

The End!  
謝謝大家～