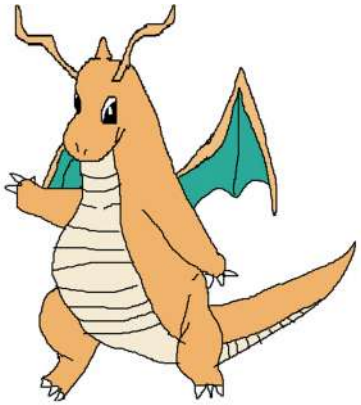
Struct



實作一個簡易的寶可夢 structure 和 function，並寫出簡易的 pseudo code (圖 1)。一隻寶可夢總共有三個基本資訊 (名字 name[32]、屬性 type、能力值 CP，CP 和 type 用 int 表示)，而 structure 還包含一個 pointer 指向寶可夢的進化型，若是無進化型則 pointer 指向 NULL。而 function Evolution 則會印出某隻寶可夢的完整進化過程 (ex:傳入小火龍，則印出小火龍、火恐龍、噴火龍)。

[圖 1]請按照下列格式填入 pseudo code

```
#include <stdio.h>

struct pokemon
{
    //four members in structure


};typedef struct pokemon Pokemon;

void Evolution(Pokemon *A)
{
    //print the answer


}
```

Linked List

給定以下的一些 include、struct 以及 main function

```c
#include <stdio.h>
#include <stdlib.h>

struct node {
    int data;
    struct node *next;
};

typedef struct node Node;
```

```c
int main(void) {
    Node *head = NULL;
    int val;
    while (scanf("%d", &val) != EOF) {
        head = insert(head, val);
    }
    print(head);
    return 0;
}
```

請用以下亂排過的 code(左)拼湊出原本(右)的樣子吧！

```c
    head = head->next;
    print(head->next);
    printf("%d ", head->data);
    printf("\n");
Node *n = (Node*)malloc(sizeof(Node));
else {
if (head == NULL)
n->data = d;
n->next = head;
return head;
return n;
while (head != NULL && head->data != d) {
}
}

```

```c
Node* insert(Node *head, int d) {
    // A
    // B
    // C
    // D
}

void print(Node *head) {
    // E
    // F
    // G
    // H
    // I
    // J
}

Node* find(Node *head, int d) {
    // K
    // L
    // M
    // N
}
```

Pointer

1. What will be the output of this program?

(A)
s = Hello World!
s = Hello World!
str = Hello World!QQ

(B)
s = Goodbye World!
s = Goodbye World!QQ
str = Goodbye World!QQ

(C)
s = Hello World!
s = Hello World!
[segmentation fault]

(D)
s = Goodbye World!
s = Goodbye World!QQ
[segmentation fault]

```c
#include <stdio.h>
#include <string.h>

void func_one(char *str) {
  char source[20] = "Goodbye World!";
  strcpy(str, source);
}

char* func_two(char str[]) {
  int i = strlen(str);
  str[i++] = 'Q';
  str[i++] = 'Q';
  str[i++] = '\0';
  return str;
}

int main() {
  char s[20] = "Hello World!";

  func_one(s);
  printf("s = %s\n", s);

  char* str = func_two(s);
  printf("s = %s\n", s);
  printf("str = %s\n", str);

  return 0;
}
```

Loop

```c
#include <stdio.h>

int table[4] = {1, 2, 3, 4};

int exists_in_table(int v) {
    for(int i = 1; i <=4; i++)
        if(table[i] == v)
            return 1;
    return 0;
}

int main(){
    printf("%d\n", exists_in_table(0));
    return 0;
}
```

1. What's the output? Does it make sense?
2. Is there any bug in this code? If so, fix it.

Recursion

下圖程式碼中的 fib() 函式匯回傳斐波那契數列指定項的值，如:
fib(0) = 0， fib(1) = 1， fib(2) = 1，以此類推。

```
int fib(int n) {
    if(  (1)  )
        return 0;
    if(  (2)  )
        return 1;
    return fib(n-1) + fib(n-2);
}
```

1. 請填入空缺的部分。
2. 請問如果呼叫 fib(7) 總共會執行幾個 fib() 函式？

答案

1. Struct

```c
struct pokemon
{
    char name[32];
    int type;
    int CP;
    struct pokemon *next;
}; typedef struct pokemon Pokemon;

void Evolution(Pokemon *A)
{
    while(A->next!=NULL){
        printf("%s\n",A->name);
        A = A->next;
    }
    printf("%s\n",A->name);
    return;
}
```

---

2. Linked list

```c
Node* insert(Node *head, int d) {
    Node *n = (Node*)malloc(sizeof(Node));
    n->data = d;
    n->next = head;
    return n;
}

void print(Node *head) {
    if (head == NULL)
        printf("\n");
    else {
        printf("%d ", head->data);
        print(head->next);
    }
}

Node* find(Node *head, int d) {
    while (head != NULL && head->data != d) {
        head = head->next;
    }
    return head;
}

```

3. Pointer

B

---

4. Loop
(1)

1

The function is to check whether the given value exists in the global array. Since the array doesn't contain 0, the output doesn't make sense.

(2)

```c
#include <stdio.h>

int table[4] = {1, 2, 3, 4};

int exists in table(int v) {
    for(int i = 0; i < 4; i++)
        if(table[i] == v)
            return 1;
    return 0;
}

int main(){
    printf("%d\n", exists_in_table(0));
    return 0;
}
```

---

5. Recursion
(1) n <= 0

(2) n == 1

執行 41 次