《程式設計》

一、關於下列C程式碼，請說明程式執行後，程式碼編號27~33的輸出，以及其運算邏輯。（25分）

```
01  #include <stdio.h>
02  #define SIZE 30
03  typedef enum direction {North, South, East=3, West} dir_t;
04  int f1(int a, int b) {
05        int x = 3.0/a;
06        double y = (a/2)*(b%3) + x;
07        return y;
08  }
09  int f2(dir_t d) {
10        d= (North+East)/2 > d? East: West;
11        return d;
12  }
13  int f3(int a, int b) {
14        if (b==a || b<=1) return a+b;
15        else if (a<=1) return b-a;
16        else return f3(a-b, a-1)+b+a;
17  }
18  int f4(int a, int b) {
19        int data[SIZE];
20        for (int i=1, k=0; i<a; i++) {
21              if (i%2==0) data[k++]=i;
22        }
23        return data[b];
24  }
25  unsigned int f5(unsigned int a, unsigned int b) { return (~a&b);   }
26  int main() {
27        printf("%d\n", f1(10, 4));
28        printf("%d\n", f2(South));
29        printf("%d\n", f3(6, 4));
30        printf("%u\n", f3(7, 4));
31        printf("%d\n", f4(20, 5));
32        printf("%d\n", f4(10, 4));
33        printf("%u\n", f5(4, 7));
34        return 0;
35  }
```

試題評析	此試題旨於測驗學生對於函式定義、條件語句、遞迴、陣列操作及位元運算的瞭解和應用能力。試題涵蓋了多個主題，包括數學運算、邏輯判斷、遞迴調用、數組處理和位元運算，是一道綜合性強、難度較低的題目。重點在於理解每個函式的邏輯、參數的傳遞方式以及返回值的類型。此外，函式中的特殊操作，如遞迴和位元運算，需要有清晰的理解。這道題目適合用於評估學生對C語言核心概念的掌握程度。
考點命中	《高點・高上程式設計講義》第一回，Vincent編撰，頁28。

答：

(一)輸出結果為

5
4
15
25

12
10
3

(二) 函式 f1(int a, int b)：此函式進行浮點和整數的混合運算。計算過程中，3.0 / a 的結果會被隱式轉換為整數，而 a / 2 也是整數除法。函式返回 (a/2) * (b%3) 與 x 的和，結果為整數。f1(10, 4) 返回 (10/2) * (4%3) + 0，即 5 * 1 + 0，輸出為 5。

(三) 函式 f2(dir_t d)：這個函式基於 enum 類型 dir_t 的值進行條件判斷。計算 (North + East) / 2 即 (0 + 3) / 2，結果為 1.5，向下取整為 1。比較 1 與傳入的 d，返回 East 或 West。f2(South) 即 f2(1)，因為 1 > 1 為假，所以返回 West，輸出為 4（West 的值）。

(四) 函式 f3(int a, int b)：這是一個遞迴函式，依據不同的條件進行遞迴調用或返回計算結果。f3(6, 4) 和 f3(7, 4) 的輸出依賴於遞迴過程的具體計算結果。藉由遞迴傳遞追蹤，f3(6, 4) 的結果為 15、f3(7, 4) 的結果為 25。

(五) 函式 f4(int a, int b)：此函式使用一個 local 陣列 data，並填充部分元素。然後返回 data[b] 的值。但這裡存在潛在的陣列越界問題，因為 data 的填充依賴於 a 的值。f4(20, 5) 和 f4(10, 4) 的輸出取決於 data 數組在相應索引處的值，因為剛好皆未發生越界問題，f4(20, 5) 的值為 12、f4(10, 4) 的值為 10。

(六) 函式 f5(unsigned int a, unsigned int b)：這個函式進行位元運算。~a 是 a 的位元反轉，與 b 進行按位與操作。對於 f5(4, 7) 的呼叫，計算如下：4 的二進位表示為 0100。~4 的二進位表示為所有位元取反，即 1011。7 的二進位表示為 0111。因此 ~4 & 7 的結果是 1011 & 0111 = 0011，即十進位的 3。

二、針對下列C++程式，請標示出Except 類別的f1, …, f6 函式中有問題的函式，與說明其問題之原因；並請說明若將有問題的函式和程式碼刪除，其程式執行後之輸出。（25分）

```
01 #include <stdexcept>
02 #include <iostream>
03 #include <string>
04 using namespace std;
05 class Except{
06 public:
07       void f1(int c);
08       void f2();
09       void f3();
10       void f4();
11       void f5();
12       void f6();
13 };
14 int main() {
15       Except e;
16       e.f1(1);
17       e.f2();
18       e.f3();
19       e.f4();
20       e.f5();
21       e.f6();
22       return 0;
23 }
24 void Except::f1(int c) {
25       if (c<0)
26             throw out_of_range("large");
27       cout<<"exc1"<<endl;
28 }
29 void Except::f2() {
30       f1(-1);
31 }
32 void Except::f3() {
33       try {
34             f1(-1);
35             cout<<"ok"<<endl;
36       }catch(exception &e) {
37             cout<<"exc2"<<endl;
38       }
39 }
40 void Except::f4() {
41       try {
42             throw out_of_range("no");
43       }catch(out_of_range &e) {
44             cout<<e.what()<<endl;
45             cout<<"exc3"<<endl;
46       }
47 }
48 void Except::f5() {
49       try {
50             throw out_of_range("yes");
51       }catch(exception &e) {
52             cout<<"exc41"<<endl;
53       }catch(out_of_range &e) {
54             cout<<e.what()<<endl;
55             cout<<"exc42"<<endl;
56       }
57 }
58 void Except::f6() {
59       try {
60             throw out_of_range("ok");
61       }finally {
62            cout<<"exc6"<<endl;;
63       }
64 }
```

試題評析	此試題主要考學生對於 C++ 程式語言中異常處理機制的理解和應用。它涉及了異常拋出(throw)、異常捕捉 (try-catch) 以及函式的基本使用。這需要學生對 C++ 中的異常處理機制有深入的理解，特別是異常類型的匹配、try-catch 塊的工作方式以及函式的調用流程。此題目的難度適中，適合用於評估學生對 C++ 異常處理機制的掌握程度及問題分析能力。
考點命中	《高點・高上程式設計講義》第一回，Vincent編撰，頁91。

答：

(一) 有問題的函式及其問題原因：

甲、 f2()：此函式會呼叫 f1(-1)，由於 -1 小於 0，因此會拋出一個未被處理的 out_of_range 異常。在 C++ 中，如果一個異常未被捕捉，程序將會終止執行。因此，f2() 的呼叫將導致程序提早終止。

乙、 f5()：此函式中的 catch 塊順序有誤。由於 out_of_range 繼承自 exception，因此第一個 catch 塊（捕捉 exception 類型）會捕捉到應該由第二個 catch 塊（專門捕捉 out_of_range 類型）處理的異常。這會導致第二個 catch 塊中的代碼永遠不會執行。

丙、 f6()：C++ 沒有 finally 這個關鍵字。儘管 f6() 中的意圖是無論是否捕獲異常都執行某些代碼，但是由於語法錯誤，這會導致編譯失敗。

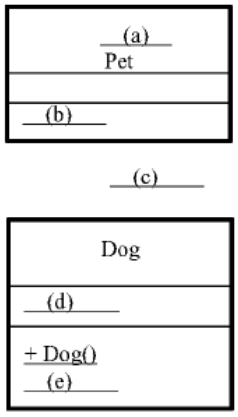
(二) 修改後的執行輸出結果：

移除了 f2() 函式和其在 main 函式中的呼叫，調換了 f5() 中的 catch 塊順序，並修正了 f6() 函式。輸出結果為：

```
excl
exc2
no
exc3
exc41
exc6
```

三、針對下列Java程式碼，請完成統一塑模語言（UML）類別圖(a)~(e)；另外請標示出錯誤程式碼行數並說明錯誤原因；以及說明若將錯誤行數程式碼予以註解後，執行其程式的輸出。（25分）

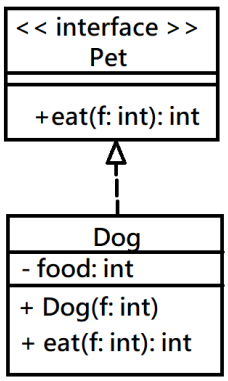
```
01  import java.io.*;
02  interface Pet {
03      public abstract int eat(int f);
04  };
05  class Dog implements Pet {
06      public Dog(int f) {food = f; }
07      public int eat(int f) {
08          food += f;
09          return food;
10      }
11      private int food;
12  };
13  public class Main{
14      public static void main(String[] args) {
15          Pet d1 = new Pet();
16          Pet d2 = new Dog();
17          Pet d3 = new Dog(5);
18          d1.eat(5);
19          d2.eat(5);
20          System.out.println("dog: "+d3.eat(5));
21      }
22  }
```



試題評析	此試題旨在考學生對於Java程式語言中的物件導向概念、介面(Interface)使用、類別(Class)的實作，以及基本錯誤診斷的能力，並能運用UML類別圖來表示程式結構。此外，該題目也考驗學生對於繼承和多型特性的理解。題目的難度簡單，適合用於評估學生對Java物件導向程式設計、錯誤診斷和修正，以及UML類別圖應用的掌握程度。
考點命中	《高點・高上程式設計講義》第一回，Vincent編撰，頁73。

答：

(一) UML 類別圖



(二) 第10行：Pet d1 = new Pet(); 是錯誤的，因為Pet是一個介面，不能被實例化。
第11行：Pet d2 = new Dog(); 缺少必要的建構參數。應該是Pet d2 = new Dog(初始食物量);。
若將錯誤行數程式碼註解後（第10行和第11行），則程序的執行輸出將是：
dog: 10

四、針對下列Python程式碼，依序在兩個Terminal執行server.py和client.py後，在client.py輸入Tom和quit；請說明client.py的Terminal之輸出內容，並說明Line 03,04,05程式碼的運作邏輯。（25分）

```
01  # server.py
02  import socket
03  s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
04  s.bind(('127.0.0.1', 7000))
05  s.listen(5)
06  print('wait for connection...')
07  while True:
08      conn, addr = s.accept()
09      print('connected by ' + str(addr))
10      indata = conn.recv(1024)
11      print('recv: ' + indata.decode())
12      if 'quit' in indata.decode():
13          outdata = 'bye '
14      else:
15          outdata = 'hi ' + indata.decode()
16      conn.send(outdata.encode())
17      conn.close()
18      if 'quit' in indata.decode():
19          break
20      print('listen...')
21  s.close()
22
23
24  #client.py
25  import socket
26  while True:
27      s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
28      s.connect(('127.0.0.1', 7000))
29      name = input('>name:')
30      print('send: ' + name)
31      s.send(name.encode())
32      indata = s.recv(1024)
33      s.close()
34      print('>' + indata.decode())
35      if 'quit' in name:
36          break
37
38
39
40
```

試題評析	此試題主要考學生對Python程式語言中網路程式的基礎知識與應用。特別是針對socket編程的理解，包括建立伺服器(server)和客戶端(client)的通訊，以及如何通過socket傳輸數據。試題涵蓋了socket的建立、連接、數據傳輸和關閉連接等關鍵步驟。由於程設往年並未特別著墨於此領域，該題目的難度較高，將具有一定的鑑別度。
考點命中	《高點・高上程式設計講義》第一回，Vincent編撰，頁77。

答：

(一) 當在 client.py 輸入 "Tom" 和 "quit" 後，會分別向伺服器發送這些數據，並接收伺服器的回應。

甲、 輸入 "Tom" 時，client.py 會向伺服器發送字符串 "Tom"，伺服器接收到後，會回應 "hi Tom"。所以 client.py 的輸出將會是：

```
>name:Tom
send: Tom
>hi Tom
```

乙、 輸入 "quit"，client.py 會向伺服器發送 "quit"，伺服器接收到 "quit" 之後，會回應 "bye"，並關閉連接。client.py 的輸出將會是：

```
>name:quit
send: quit
>bye
```


(二) Line03: s = socket.socket(socket.AF_INET, socket.SOCK_STREAM) 建立一個 socket 物件。socket.AF_INET 表示使用 IPv4 協議，socket.SOCK_STREAM 表示使用 TCP 協議。

Line04: s.bind(('127.0.0.1', 7000)) 將 socket 綁定到本地地址 127.0.0.1 和端口 7000。這是伺服器監聽客戶端請求的地址和端口。

Line05: s.listen(5) 使 socket 進入等待客戶端連接的狀態，參數 5 表示最大的等待連接數。