C++程式數計：參考

目錄

1. 基本概念

參考(reference)可以視為別名（alias）

- 一個變數在宣告時只有一個名字，就像一個人只有一個身分證名字；但一個變數可以有多個參考別名（就像一個人可能有很多綽號或代號）。
- 參考變數就是原變數的別名，等於是另一個可以操作同一塊記憶體的變數名稱。在 C++ 中，別名和原名幾乎沒有差別，可以直接使用。

參考變數也可以視為「類似固定指標」

- 可以將參考變數想像成「目前指向某變數的指標」，但不能改變所指向的變數(不同於指標（pointer），指標可以隨時指向不同的變數，參考變數一旦綁定後就不能變更目標)
- 因此，參考變數在宣告時必須初始化，不能像指標一樣先宣告再指向(語法上使用參考不需要加 * 或 &，比指標更直覺簡潔。)

2. 宣告及初始化

```
int i;
int &j = i;  // j 是 i 的參考
int &k = j;  // k 也是 i 的參考（間接）
// 在變數名稱前加上 & 表示該變數為參考變數
// 參考變數必須初始化，因為它一旦與某個變數綁定後，就不能再指向其他變數。
// j 和 k 是 i 的別名（alias），都共享相同記憶體位置。
```

2.1 範例：參考變數範例

```
#include <iostream>
using namespace std;

int main(void) {
    int i, j;
    int &ref1 = i;
    int &ref2 = ref1;

    // cin >> i;
    i = 5;
    cout << "i = " << i << endl; // 5
    cout << "&i = " << &i << endl; // 0x7ffffcbec
    cout << "&j = " << &j << endl; // 0x7ffffcbe8
    cout << "ref1 = " << ref1 << endl; // 5
    cout << "ref2 = " << ref2 << endl; // 5

    ref1 = 8;
    j = ref2 + 3;
    cout << "j = " << j << endl;
}
```

3. 參考中的參數傳遞

3.1 範例：指標參數傳遞

```
#include <iostream>
using namespace std;

void pointerInc(double* &p1, double* &p2){
    cout << "pointerInc starts" << endl;
    cout << "The address of p1 is " << &p1 << endl;
    cout << "The value of p1 is " << p1 << endl;
    cout << "The address of p2 is " << &p2 << endl;
    cout << "The value of p2 is " << p2 << endl;

    *p1 += 1;
    p1 = p2;
    *p1 += 2;
    cout << "The value of p1 is " << p1 << endl;
    cout << "The value of p2 is " << p2 << endl;
```

```
    cout << "pointerInc ends" << endl;
}

int main(void){
    double i, j;
    double *iptr {&i};
    double *jptr {&j};

    // cin >> i >> j;
    i = 5;
    j = 6;

    cout << "The address of i is " << &i << endl;
    cout << "The value of i is " << i << endl;
    cout << "The address of j is " << &j << endl;
    cout << "The value of j is " << j << endl;
    cout << "The address of iptr is " << &iptr << endl;
    cout << "The value of iptr is " << iptr << endl;
    cout << "The address of jptr is " << &jptr << endl;
    cout << "The value of jptr is " << jptr << endl;

    pointerInc(iptr, jptr);
    cout << "after pointerInc " << endl;
    cout << "The value of i is " << i << endl;
    cout << "The value of j is " << j << endl;

    *iptr += 5;
    cout << "after *iptr += 5" << endl;
    cout << "The value of iptr is " << iptr << endl;
    cout << "The value of jptr is " << jptr << endl;
    cout << "The value of i is " << i << endl;
    cout << "The value of j is " << j << endl;
    return 0;
}
```

3.2 範例：藉由參考交換變數值

```
#include <iostream>
using namespace std;

void swap(int &a, int &b){
```

```
    int temp;
    temp = a;
    a = b;
    b = temp;
}
int main(void){
    int i, j;
    cin >> i >> j;
    cout << "i = " << i << " j = " << j << endl;
    swap(i, j);
    cout << "i = " << i << " j = " << j << endl;
    return 0;
}
```

4. 參考回傳值

參考也可以作為回傳值，作法是在函數前面加一個&

4.1 範例：傳回第一個正整數

```
#include <iostream>
using namespace std;

int &firstPositive(int *ptr){
    while (*ptr <= 0){
        ptr++;
    }
    return *ptr;
}
int main(void){
    int array[] {0, 0, 3, 0, 5, 0, 0, 10};
    int &iref {firstPositive(array)};
    cout << "iref = " << iref << endl;
    iref = 0;

    int i {firstPositive(array)};
    cout << "i = " << i << endl;
    i = 0;
    cout << "array[4] = " << array[4] << endl;

    i = firstPositive(array);
    cout << "i = " << i << endl;
```

```
    firstPositive(array) = 0;

    i = firstPositive(array);
    cout << "i = " << i << endl;
    iref = 7;

    i = firstPositive(array);
    cout << "i = " << i << endl;
    (*(&iref - 1))++;

    i = firstPositive(array);
    cout << "i = " << i << endl;
    return 0;
}
```

4.2 範例：讓所有元素相加

```
#include <iostream>
using namespace std;
#define N 6

int main(){
    int keys[N];
    int sum {0};
    for (int &v : keys)
        cin >> v;
    for (int &v : keys)
        sum += v;
    cout << sum << endl;
    return 0;
}
```