

ZeroJudge b511 換銅板

<https://zerojudge.tw/ShowProblem?problemid=b511>

Pei-yih Ting

112/11/20

題目說明

- 輸入 n 個不同面值的銅板 $\{d_1, d_2, \dots, d_n\}$ ，然後輸入一個金額 v ，將全部可能的找零方式列出。譬如有 3 種銅板面值分別是 1 元、5 元、10 元，假設要湊出 17 元，如果把找零方法表示成“(1 元個數, 5 元個數, 10 元個數)”，共有右列幾種方法：

題目說明

- 輸入 n 個不同面值的銅板 $\{d_1, d_2, \dots, d_n\}$ ，然後
輸入一個金額 v ，將全部可能的找零方式列出。
譬如有 3 種銅板面值分別是 1 元、5 元、10 元，
假設要湊出 17 元，如果把找零方法表示成“(1
元個數, 5 元個數, 10 元個數)”，共有右列幾種
方法：

(2,1,1)

(2,3,0)

(7,0,1)

(7,2,0)

(12,1,0)

(17,0,0)

題目說明

- 輸入 n 個不同面值的銅板 $\{d_1, d_2, \dots, d_n\}$ ，然後
輸入一個金額 v ，將全部可能的找零方式列出。
譬如有 3 種銅板面值分別是 1 元、5 元、10 元，
假設要湊出 17 元，如果把找零方法表示成“(1
元個數, 5 元個數, 10 元個數)”，共有右列幾種
方法：

(2,1,1)

(2,3,0)

(7,0,1)

(7,2,0)

(12,1,0)

(17,0,0)

$$1 \leq n \leq 5 \quad 1 \leq d_i, v \leq 100$$

題目說明

- 輸入 n 個不同面值的銅板 $\{d_1, d_2, \dots, d_n\}$ ，然後
輸入一個金額 v ，將全部可能的找零方式列出。
譬如有 3 種銅板面值分別是 1 元、5 元、10 元，
假設要湊出 17 元，如果把找零方法表示成“(1
元個數, 5 元個數, 10 元個數)”，共有右列幾種
方法：

(2,1,1)

(2,3,0)

(7,0,1)

(7,2,0)

(12,1,0)

(17,0,0)

$$1 \leq n \leq 5 \quad 1 \leq d_i, v \leq 100$$

- 排列順序的規則: lexicographic order, 例如 (7,2,0) 先於
(12,1,0) 因為 7 比 12 小; 而 (7,0,1) 和 (7,2,0) 的順序，因為第
一個數目 7 和 7 相等，這時候就要比第二個數，而 0 小於
2 所以 (7,0,1) 先於 (7,2,0)。

Baseline – deep *for* loops

Baseline – deep *for* loops

```
int main() {
    int i, n, d[5], m[5], c[5], v, V[5];
    while (1==scanf("%d", &n)) {
        for (i=0; i<n; i++)
            scanf("%d", &d[i]);
        scanf("%d", &v);
        for (i=0; i<n; i++) m[i] = v/d[i];
        // 5 levels of for loops
    }
    return 0;
}
```

Baseline – deep *for* loops

```
int main() {
    int i, n, d[5], m[5], c[5], v, V[5];
    while (1==scanf("%d", &n)) {
        for (i=0; i<n; i++)
            scanf("%d", &d[i]);
        scanf("%d", &v);
        for (i=0; i<n; i++) m[i] = v/d[i];
        // 5 levels of for loops
    }
    return 0;
}
```

```
void print(int n, int c[]) {
    for (int i=0; i<n; i++)
        printf("%c%d", "[i>0], c[i]);
    printf("\n");
}
```


Baseline – deep *for* loops

```
int main() {
    int i, n, d[5], m[5], c[5], v, V[5];
    while (1==scanf("%d", &n)) {
        for (i=0; i<n; i++)
            scanf("%d", &d[i]);
        scanf("%d", &v);
        for (i=0; i<n; i++) m[i] = v/d[i];
        // 5 levels of for loops
    }
    return 0;
}

void print(int n, int c[]) {
    for (int i=0; i<n; i++)
        printf("%c%d", "(,[i>0], c[i]);
    printf("\n");
}
```

```
for (V[0]=c[0]=0; c[0]<=m[0]; c[0]++,V[0]+=d[0]) {
```

```
    for (V[1]=V[0],c[1]=0; c[1]<=m[1]; c[1]++,
        V[1]+=d[1]) {
```

```
        for (V[2]=V[1],c[2]=0; c[2]<=m[2]; c[2]++,
            V[2]+=d[2]) {
```

```
            for (V[3]=V[2],c[3]=0; c[3]<=m[3]; c[3]++,
                V[3]+=d[3]) {
```

```
                for (V[4]=V[3],c[4]=0; c[4]<=m[4]; c[4]++,
                    if (V[4]==v) print(n,c);          V[4]+=d[4])
```

```
                }
```

```
            }
```

```
        }
```

```
    }
```

如果 n==5

Baseline – deep *for* loops

```
int main() {
    int i, n, d[5], m[5], c[5], v, V[5];
    while (1==scanf("%d", &n)) {
        for (i=0; i<n; i++)
            scanf("%d", &d[i]);
        scanf("%d", &v);
        for (i=0; i<n; i++) m[i] = v/d[i];
        // 5 levels of for loops
    }
    return 0;
}

void print(int n, int c[]) {
    for (int i=0; i<n; i++)
        printf("%c%d", "(,[i>0], c[i]);
    printf("\n");
}
```

```
for (V[0]=c[0]=0; c[0]<=m[0]; c[0]++,V[0]+=d[0]) {
```

```
    for (V[1]=V[0],c[1]=0; c[1]<=m[1]; c[1]++,
        V[1]+=d[1]) {
```

```
        for (V[2]=V[1],c[2]=0; c[2]<=m[2]; c[2]++,
            V[2]+=d[2]) {
```

```
            for (V[3]=V[2],c[3]=0; c[3]<=m[3]; c[3]++,
                if (n==4) { V[3]+=d[3]) {
```

```
                    if (V[3]==v) print(n,c); 如果 n==4
                    continue; }
```

```
                    for (V[4]=V[3],c[4]=0; c[4]<=m[4]; c[4]++,
                        if (V[4]==v) print(n,c); V[4]+=d[4])
```

```
                }
```

```
            }
```

```
        }
```

```
    }
```

Baseline – deep *for* loops

```
int main() {
    int i, n, d[5], m[5], c[5], v, V[5];
    while (1==scanf("%d", &n)) {
        for (i=0; i<n; i++)
            scanf("%d", &d[i]);
        scanf("%d", &v);
        for (i=0; i<n; i++) m[i] = v/d[i];
        // 5 levels of for loops
    }
    return 0;
}

void print(int n, int c[]) {
    for (int i=0; i<n; i++)
        printf("%c%d", "(", "[i>0], c[i]);
    printf("\n");
}
```

```
for (V[0]=c[0]=0; c[0]<=m[0]; c[0]++,V[0]+=d[0]) {
```

```
    for (V[1]=V[0],c[1]=0; c[1]<=m[1]; c[1]++,
        V[1]+=d[1]) {
```

```
        for (V[2]=V[1],c[2]=0; c[2]<=m[2]; c[2]++,
            if (n==3) {
                if (V[2]==v) print(n,c);
                continue; }
            V[2]+=d[2]) {
```

如果 n==3

```
                for (V[3]=V[2],c[3]=0; c[3]<=m[3]; c[3]++,
                    if (n==4) {
                        if (V[3]==v) print(n,c);
                        continue; }
                    V[3]+=d[3]) {
```

```
                        for (V[4]=V[3],c[4]=0; c[4]<=m[4]; c[4]++,
                            if (V[4]==v) print(n,c);
                            V[4]+=d[4])
```

```
                        }
```

```
                    }
```

```
                }
```

```
            }
```

Baseline – deep *for* loops

```
int main() {
    int i, n, d[5], m[5], c[5], v, V[5];
    while (1==scanf("%d", &n)) {
        for (i=0; i<n; i++)
            scanf("%d", &d[i]);
        scanf("%d", &v);
        for (i=0; i<n; i++) m[i] = v/d[i];
        // 5 levels of for loops
    }
    return 0;
}

void print(int n, int c[]) {
    for (int i=0; i<n; i++)
        printf("%c%d", "(,[i>0], c[i]);
    printf("\n");
}
```

```
for (V[0]=c[0]=0; c[0]<=m[0]; c[0]++,V[0]+=d[0]) {
```

```
    for (V[1]=V[0],c[1]=0; c[1]<=m[1]; c[1]++,
        if (n==2) {
            if (V[1]==v) print(n,c);
            continue; }
        for (V[2]=V[1],c[2]=0; c[2]<=m[2]; c[2]++,
            if (n==3) {
                if (V[2]==v) print(n,c);
                continue; }
            for (V[3]=V[2],c[3]=0; c[3]<=m[3]; c[3]++,
                if (n==4) {
                    if (V[3]==v) print(n,c);
                    continue; }
                for (V[4]=V[3],c[4]=0; c[4]<=m[4]; c[4]++,
                    if (V[4]==v) print(n,c);
                }
            }
        }
    }
```

如果 n==2

Baseline – deep *for* loops

```
int main() {
    int i, n, d[5], m[5], c[5], v, V[5];
    while (1==scanf("%d", &n)) {
        for (i=0; i<n; i++)
            scanf("%d", &d[i]);
        scanf("%d", &v);
        for (i=0; i<n; i++) m[i] = v/d[i];
        // 5 levels of for loops
    }
    return 0;
}

void print(int n, int c[]) {
    for (int i=0; i<n; i++)
        printf("%c%d", "(", "[i>0], c[i]);
    printf("\n");
}
```

```
for (V[0]=c[0]=0; c[0]<=m[0]; c[0]++,V[0]+=d[0]) {
    if (n==1) {
        if (V[0]==v) print(n,c);    如果 n==1
        continue; }
    for (V[1]=V[0],c[1]=0; c[1]<=m[1]; c[1]++,
        V[1]+=d[1]) {
        if (n==2) {
            if (V[1]==v) print(n,c);
            continue; }
        for (V[2]=V[1],c[2]=0; c[2]<=m[2]; c[2]++,
            V[2]+=d[2]) {
            if (n==3) {
                if (V[2]==v) print(n,c);
                continue; }
            for (V[3]=V[2],c[3]=0; c[3]<=m[3]; c[3]++,
                V[3]+=d[3]) {
                if (n==4) {
                    if (V[3]==v) print(n,c);
                    continue; }
                for (V[4]=V[3],c[4]=0; c[4]<=m[4]; c[4]++,
                    if (V[4]==v) print(n,c);    V[4]+=d[4])
                }
            }
        }
    }
}
```

Baseline – deep *for* loops

```
int main() {
    int i, n, d[5], m[5], c[5], v, V[5];
    while (1==scanf("%d", &n)) {
        for (i=0; i<n; i++)
            scanf("%d", &d[i]);
        scanf("%d", &v);
        for (i=0; i<n; i++) m[i] = v/d[i];
        // 5 levels of for loops
    }
    return 0;
}

void print(int n, int c[]) {
    for (int i=0; i<n; i++)
        printf("%c%d", "(,[i>0], c[i]);
    printf("\n");
}
```

Extremely ugly if n==20

```
for (V[0]=c[0]=0; c[0]<=m[0]; c[0]++,V[0]+=d[0]) {
    if (n==1) {
        if (V[0]==v) print(n,c);    如果 n==1
        continue; }
    for (V[1]=V[0],c[1]=0; c[1]<=m[1]; c[1]++,
        V[1]+=d[1]) {
        if (n==2) {
            if (V[1]==v) print(n,c);
            continue; }
        for (V[2]=V[1],c[2]=0; c[2]<=m[2]; c[2]++,
            V[2]+=d[2]) {
            if (n==3) {
                if (V[2]==v) print(n,c);
                continue; }
            for (V[3]=V[2],c[3]=0; c[3]<=m[3]; c[3]++,
                V[3]+=d[3]) {
                if (n==4) {
                    if (V[3]==v) print(n,c);
                    continue; }
                for (V[4]=V[3],c[4]=0; c[4]<=m[4]; c[4]++,
                    if (V[4]==v) print(n,c);    V[4]+=d[4])
                }
            }
        }
    }
```

Iterative counting

Iterative counting

```
int main() {
    int i, n, d[5], m[5], c[5], v;
    while (1==scanf("%d", &n)) { // 3
        for (i=0; i<n; i++)
            scanf("%d", &d[i]), c[i]=0; // 1 5 10
        scanf("%d", &v); // 17
        for (i=0; i<n; i++) m[i] = v/d[i];
        while (next(n,m,c))
            if (val(n,d,c)==v) {
                for (i=0; i<n; i++)
                    printf("%c%d", "([i>0], c[i]);
                printf("\n");
            }
        }
    return 0;
}
```


Iterative counting

```
int val(int n, int d[], int c[]) {  
    int s=0;  
    while (n--)  
        s+=c[n]*d[n];  
    return s;  
}
```

```
int main() {  
    int i, n, d[5], m[5], c[5], v;  
    while (1==scanf("%d", &n)) { // 3  
        for (i=0; i<n; i++)  
            scanf("%d", &d[i]), c[i]=0; // 1 5 10  
        scanf("%d", &v); // 17  
        for (i=0; i<n; i++) m[i] = v/d[i];  
        while (next(n,m,c))  
            if (val(n,d,c)==v) {  
                for (i=0; i<n; i++)  
                    printf("%c%d", "(", "[i>0], c[i]);  
                printf(")\n");  
            }  
        }  
    return 0;  
}
```

Iterative counting

```
int next(int n, int m[], int c[]) {
    int i;
    for (i=n-1; i>=0; i--)
        if (c[i]<m[i])
            { c[i]++; return 1; }
        else
            c[i] = 0;
    return 0;
}
```

```
int val(int n, int d[], int c[]) {
    int s=0;
    while (n-->0)
        s+=c[n]*d[n];
    return s;
}
```

```
int main() {
    int i, n, d[5], m[5], c[5], v;
    while (1==scanf("%d", &n)) { // 3
        for (i=0; i<n; i++)
            scanf("%d", &d[i]), c[i]=0; // 1 5 10
        scanf("%d", &v); // 17
        for (i=0; i<n; i++) m[i] = v/d[i];
        while (next(n,m,c))
            if (val(n,d,c)==v) {
                for (i=0; i<n; i++)
                    printf("%c%d", "[i>0]", c[i]);
                printf("\n");
            }
    }
    return 0;
}
```

Recursive DFS

Recursive DFS

```
int main() {  
    int i, n, d[5], c[5], v;  
    while (1==scanf("%d", &n)) { // 3  
        for (i=0; i<n; i++)  
            scanf("%d", &d[i]); // 1 5 10  
        scanf("%d", &v); // 17  
        dfs(n, d, v, c, 0);  
    }  
    return 0;  
}
```

Recursive DFS

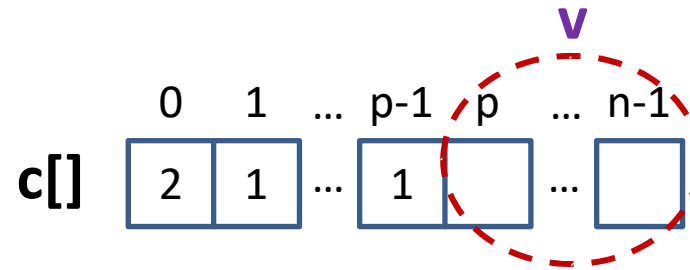
```
void dfs(int n, int d[], int v,  
        int c[], int p) {  
    int i, m;  
    if (p < n)  
        for (m = v / d[p], c[p] = 0; c[p] <= m; c[p]++, v -= d[p])  
            dfs(n, d, v, c, p + 1);  
    else  
        if (v == 0) {  
            for (i = 0; i < n; i++)  
                printf("%c%d", i > 0 ? ',': '(',  
                    c[i]);  
            printf("\n");  
        }  
}
```

```
int main() {  
    int i, n, d[5], c[5], v;  
    while (1 == scanf("%d", &n)) { // 3  
        for (i = 0; i < n; i++)  
            scanf("%d", &d[i]); // 1 5 10  
        scanf("%d", &v); // 17  
        dfs(n, d, v, c, 0);  
    }  
    return 0;  
}
```

Recursive DFS

```
void dfs(int n, int d[], int v,
        int c[], int p) {
    int i, m;
    if (p < n)
        for (m = v / d[p], c[p] = 0; c[p] <= m; c[p]++, v -= d[p])
            dfs(n, d, v, c, p + 1);
    else
        if (v == 0) {
            for (i = 0; i < n; i++)
                printf("%c%d", i > 0 ? ',': '(',
                    c[i]);
            printf("\n");
        }
}
```

v is the value to be matched to the sum of $d[p] * c[p] + \dots + d[n-1] * c[n-1]$



```
int main() {
    int i, n, d[5], c[5], v;
    while (1 == scanf("%d", &n)) { // 3
        for (i = 0; i < n; i++)
            scanf("%d", &d[i]); // 1 5 10
        scanf("%d", &v); // 17
        dfs(n, d, v, c, 0);
    }
    return 0;
}
```