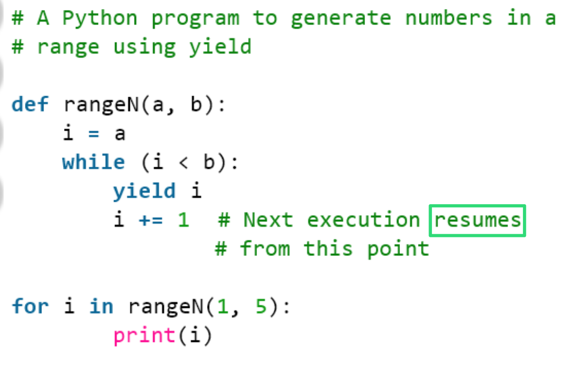


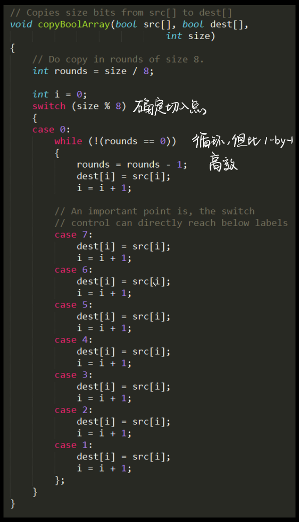
```python
# A Python program to generate numbers in a
# range using yield

def rangeN(a, b):
    i = a
    while (i < b):
        yield i
        i += 1  # Next execution resumes
                # from this point

for i in rangeN(1, 5):
        print(i)
```

Duff's Device

① unrolled loop

② use switch statement to jump into center of the ooce

一次拷贝多个值



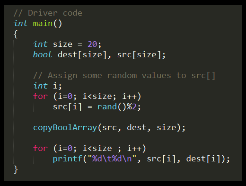
```c
// Copies size bits from src[] to dest[]
void copyBoolArray(bool src[], bool dest[],
                                int size)
{
    // Do copy in rounds of size 8.
    int rounds = size / 8;

    int i = 0;
    switch (size % 8)
    {
    case 0:
        while (!(rounds == 0))
        {
            rounds = rounds - 1;
            dest[i] = src[i];
            i = i + 1;

        // An important point is, the switch
        // control can directly reach below labels
        case 7:
            dest[i] = src[i];
            i = i + 1;
        case 6:
            dest[i] = src[i];
            i = i + 1;
        case 5:
            dest[i] = src[i];
            i = i + 1;
        case 4:
            dest[i] = src[i];
            i = i + 1;
        case 3:
            dest[i] = src[i];
            i = i + 1;
        case 2:
            dest[i] = src[i];
            i = i + 1;
        case 1:
            dest[i] = src[i];
            i = i + 1;
        };
    }
}
```

确定切入点

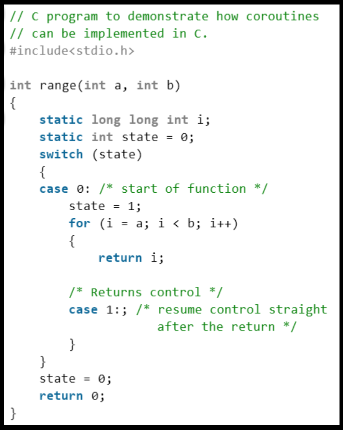
循环，但比 1-by-1 高效

逐行循环

本会在 loop 中的逻辑
现在 line-by-line



```c
// Driver code
int main()
{
    int size = 20;
    bool dest[size], src[size];

    // Assign some random values to src[]
    int i;
    for (i=0; i<size; i++)
        src[i] = rand()%2;

    copyBoolArray(src, dest, size);

    for (i=0; i<size ; i++)
        printf("%d\t%d\n", src[i], dest[i]);
}
```



```c
// C program to demonstrate how coroutines
// can be implemented in C.
#include<stdio.h>

int range(int a, int b)
{
    static long long int i;
    static int state = 0;
    switch (state)
    {
    case 0: /* start of function */
        state = 1;
        for (i = a; i < b; i++)
        {
            return i;

        /* Returns control */
        case 1:; /* resume control straight
                    after the return */
        }
    }
    state = 0;
    return 0;
}
```