- Username : class-1
- Password :

- Check VS 2019 whether can use
- We will start our course in 18:30
- we will start demonstrate the exercises at 19:15.
- Do not use scanf_s
- Please make sure the TA has recorded your exercise score here before leaving.

Loop

Introduction to Computers and Programming

2023/09/26

Outline

- Loops
  - for
  - while
  - do…while
  - break / continue
- Generate random number
- Exercise

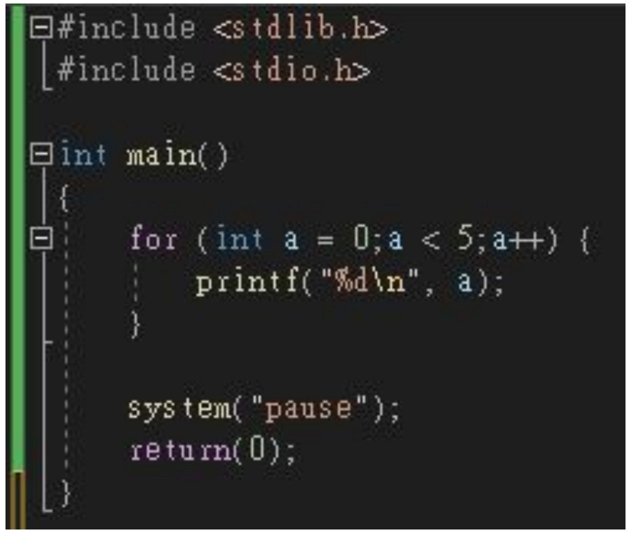
For Loop

for(initialization ; condition ; increment)

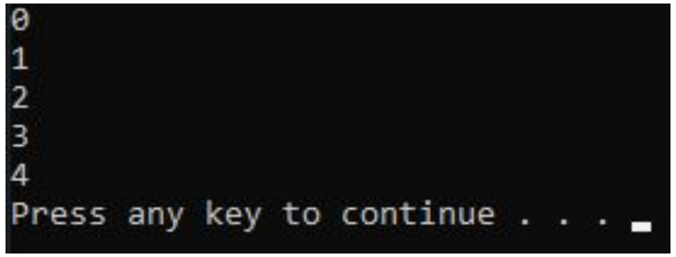
{

statements;

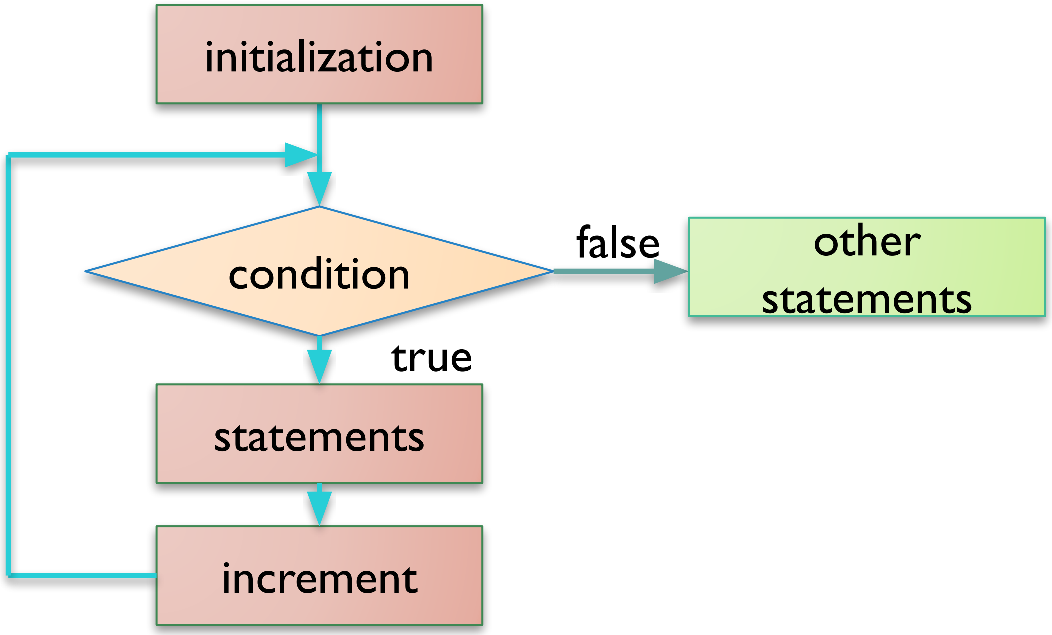
}



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

int main()
{
    for (int a = 0;a < 5;a++) {
        printf("%d\n", a);
    }

    system("pause");
    return(0);
}
```

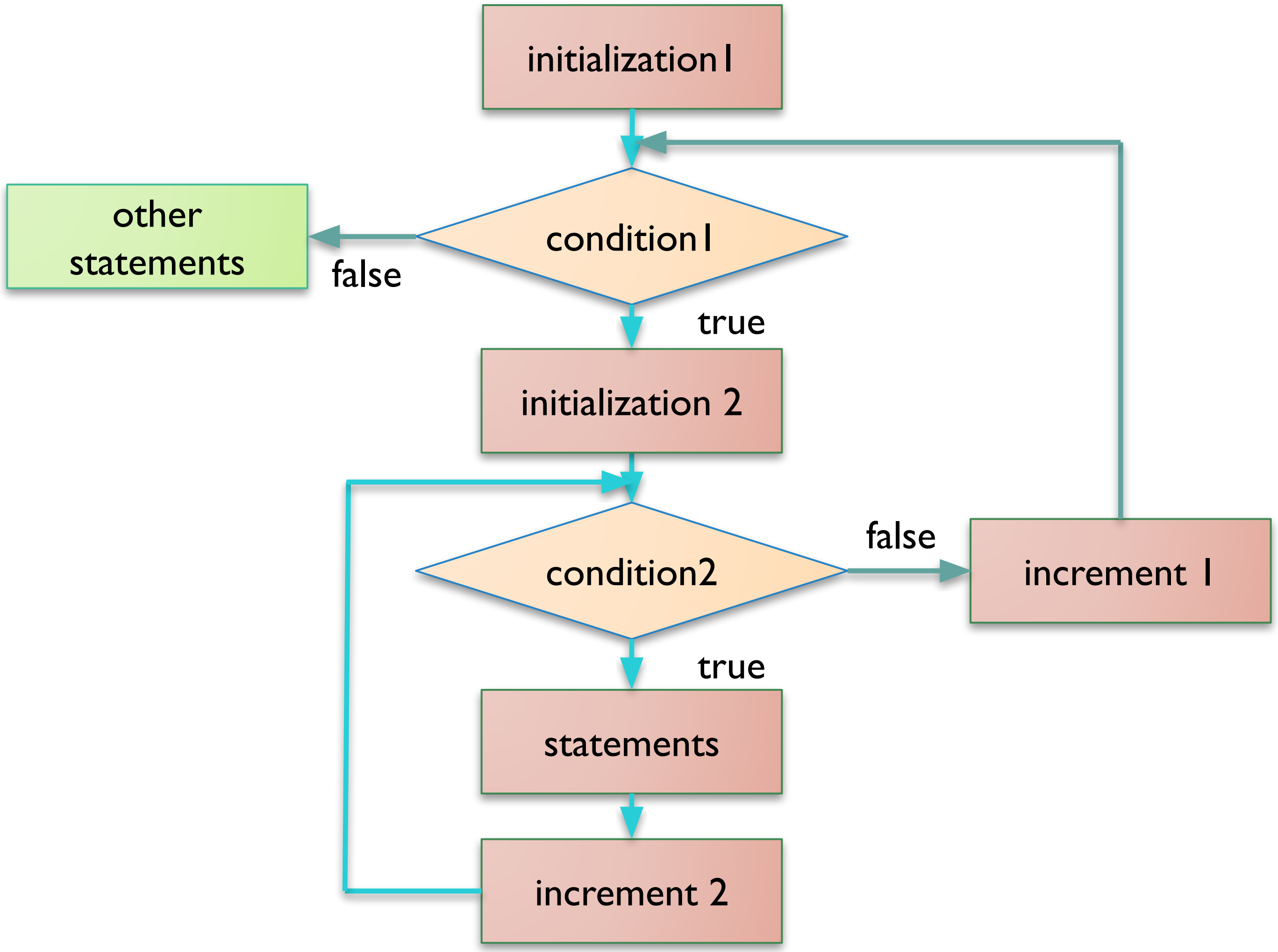


```
0
1
2
3
4
Press any key to continue . . .
```

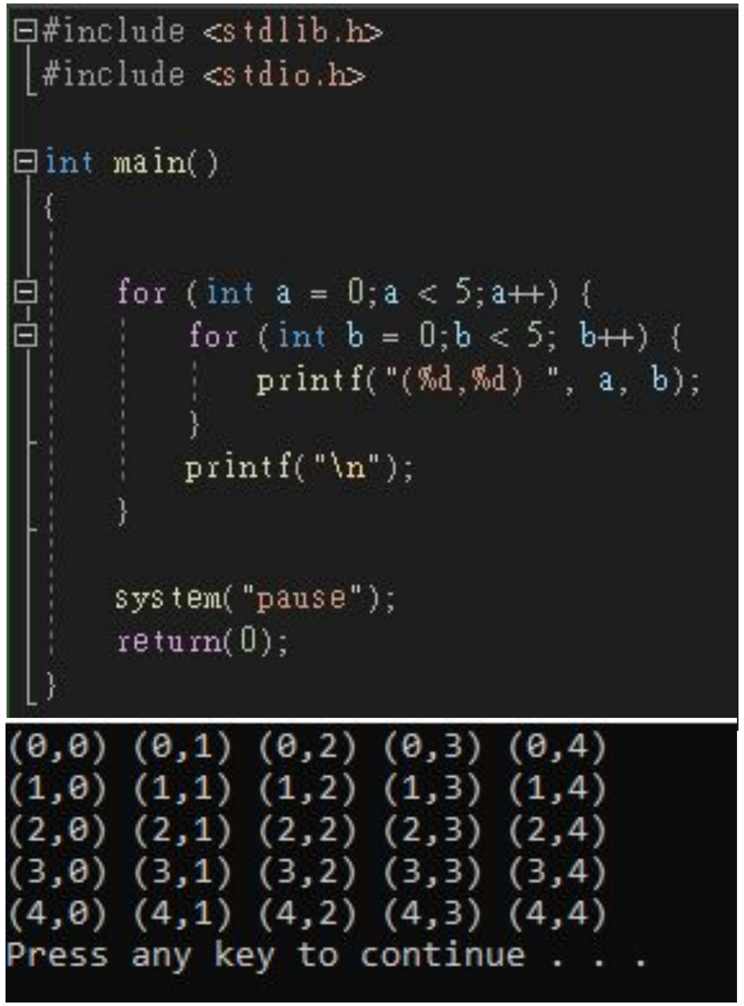


Nested For Loop

```
for(initialization1 ; condition1 ; increment1)
{
    for(initialization2 ; condition2 ; increment2)
    {
        statements;
    }
}
```

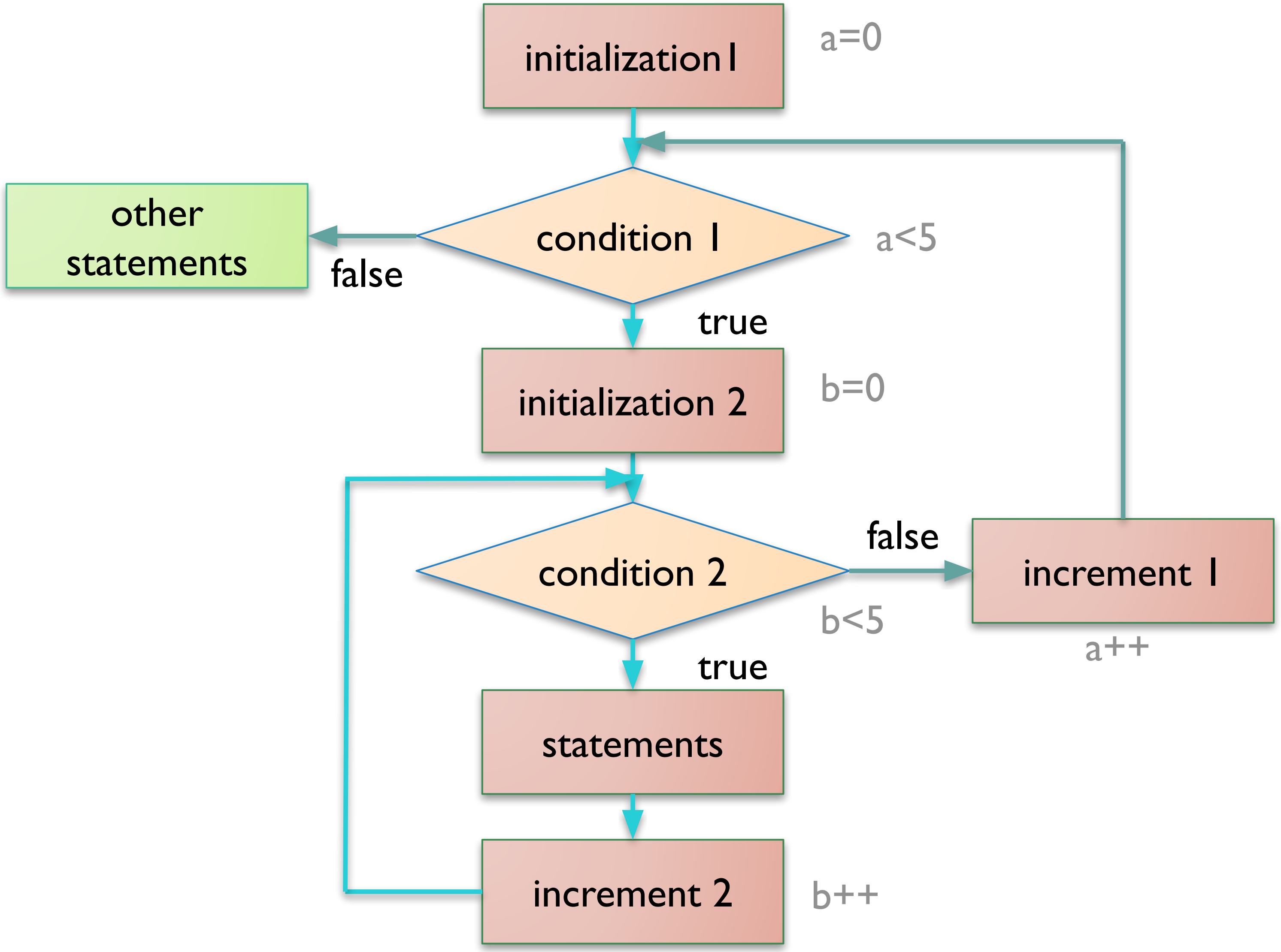


Nested For Loop



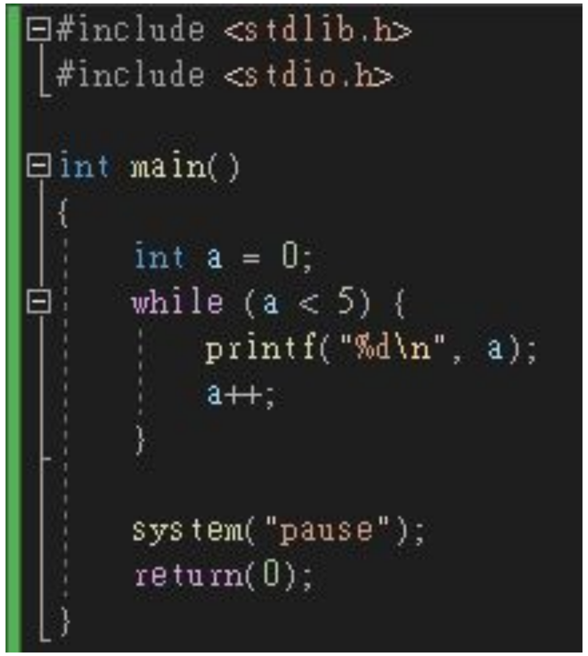
```
#include <stdlib.h>
#include <stdio.h>

int main()
{

    for (int a = 0;a < 5;a++) {
        for (int b = 0;b < 5; b++) {
            printf("(%d,%d) ", a, b);
        }
        printf("\n");
    }

    system("pause");
    return(0);
}
```

```
(0,0) (0,1) (0,2) (0,3) (0,4)
(1,0) (1,1) (1,2) (1,3) (1,4)
(2,0) (2,1) (2,2) (2,3) (2,4)
(3,0) (3,1) (3,2) (3,3) (3,4)
(4,0) (4,1) (4,2) (4,3) (4,4)
Press any key to continue . . .
```

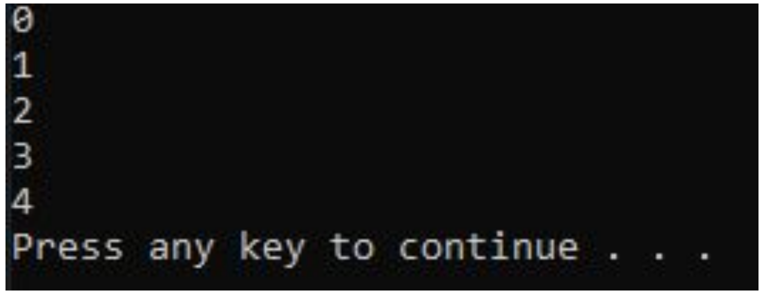


While Loop

```
while(condition)
{
    statement 1;
    statement 2;
    ....
}
```



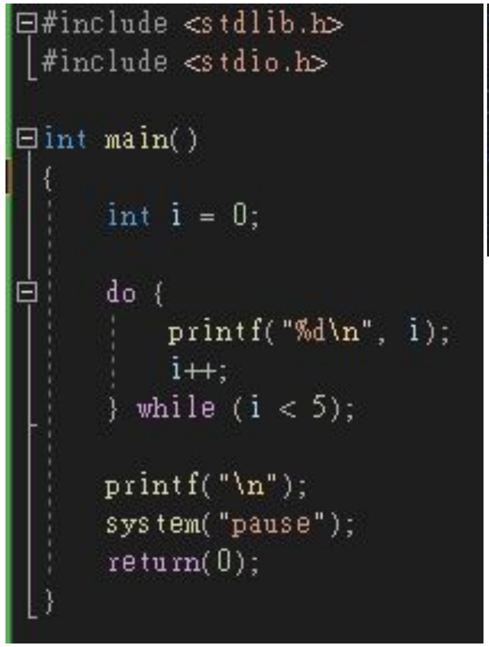
```
#include <stdlib.h>
#include <stdio.h>

int main()
{
    int a = 0;
    while (a < 5) {
        printf("%d\n", a);
        a++;
    }

    system("pause");
    return(0);
}
```



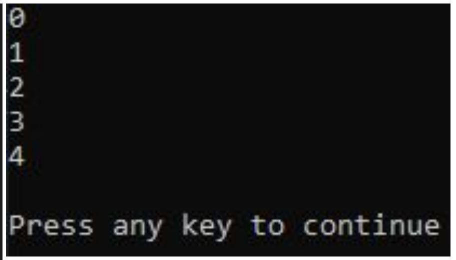
```
0
1
2
3
4
Press any key to continue . . .
```

do…while

```
do{
    statements;
}while(condition);
```

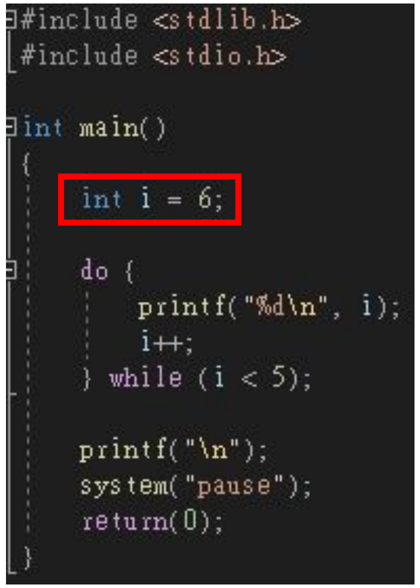


```
#include <stdlib.h>
#include <stdio.h>

int main()
{
    int i = 0;

    do {
        printf("%d\n", i);
        i++;
    } while (i < 5);

    printf("\n");
    system("pause");
    return(0);
}
```

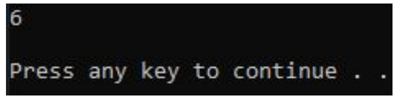


```
0
1
2
3
4

Press any key to continue
```

It is guaranteed to execute at least one time, even though it doesn’t meet the condition



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

int main()
{
    int i = 6;

    do {
        printf("%d\n", i);
        i++;
    } while (i < 5);

    printf("\n");
    system("pause");
    return(0);
}
```



```
6

Press any key to continue . .
```

break

```
for(initialization; condition;  increment)
{
    statement 1;
    statement 2;
    …
    break;
    …
    statement n
}
```

If the break statement is executed, the rest of the statements will not be executed.
Moreover, the for loop will terminate immediately.

continue

```
for(initialization; condition;  increment)
{
    statement 1;
    statement 2;
    …
    continue;
    …
    statement n
}
```

If the continue statement is executed, the statement in this block will not be executed.
But it will continue to execute the next loop.

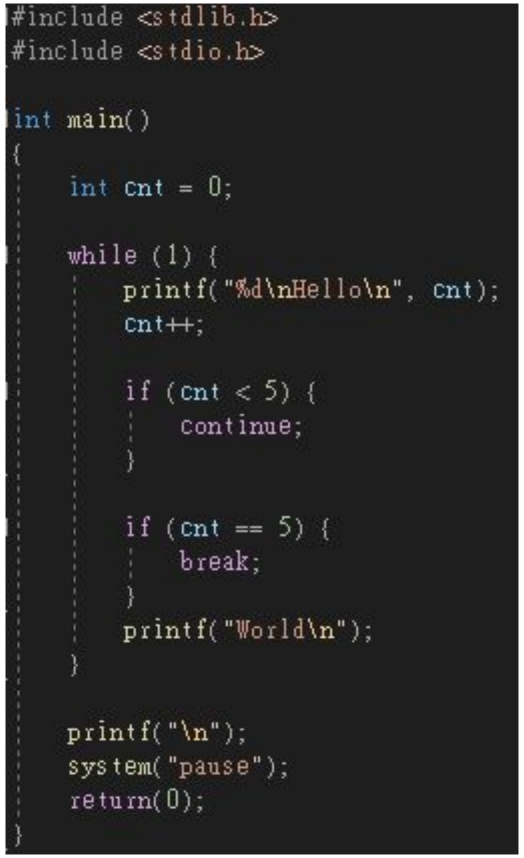
Infinite loop

Digital 1

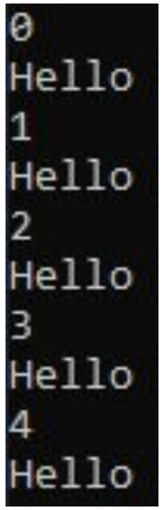
```
while(1)
{
        statement(s);
        if(condition)
        {
              statement;
              break;
         }
}
```

1. Continuous execution loop
2. Will not jump out of the loop until the condition is satisfied

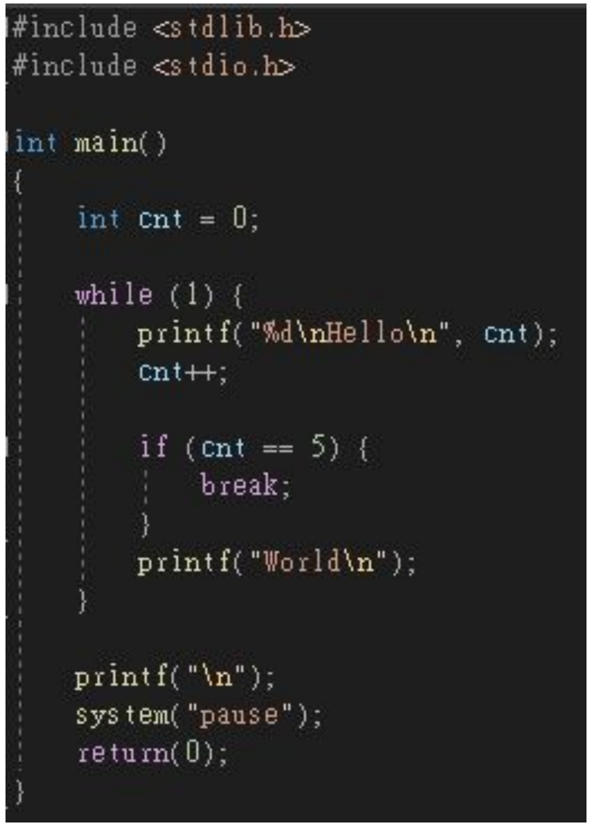
Example



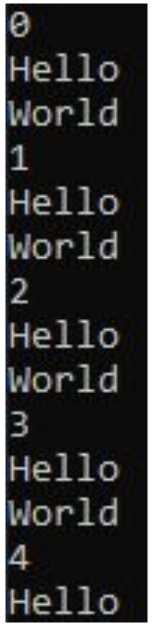
```c
#include <stdlib.h>
#include <stdio.h>

int main()
{
    int cnt = 0;

    while (1) {
        printf("%d\nHello\n", cnt);
        cnt++;

        if (cnt < 5) {
            continue;
        }

        if (cnt == 5) {
            break;
        }
        printf("World\n");
    }

    printf("\n");
    system("pause");
    return(0);
}
```



```
0
Hello
1
Hello
2
Hello
3
Hello
4
Hello
```



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

int main()
{
    int cnt = 0;

    while (1) {
        printf("%d\nHello\n", cnt);
        cnt++;

        if (cnt == 5) {
            break;
        }
        printf("World\n");
    }

    printf("\n");
    system("pause");
    return(0);
}
```



```
0
Hello
World
1
Hello
World
2
Hello
World
3
Hello
World
4
Hello
```

Generate random number

Generate an int type random number in the range [0,n)

For example : 0, 3, 9, 7, 5, 2, 1, 8, 5…

```
int rand_num = rand() % n;     // it generates a random in range [0, n)
```

Disadvantage:

In compile time, rand seed will be determined, so return value are all the same.

Generate random number

- srand(unsigned int seed)
  - according seed to update rand() return value.
- time_t time(time_t *t)
  - return timestamp

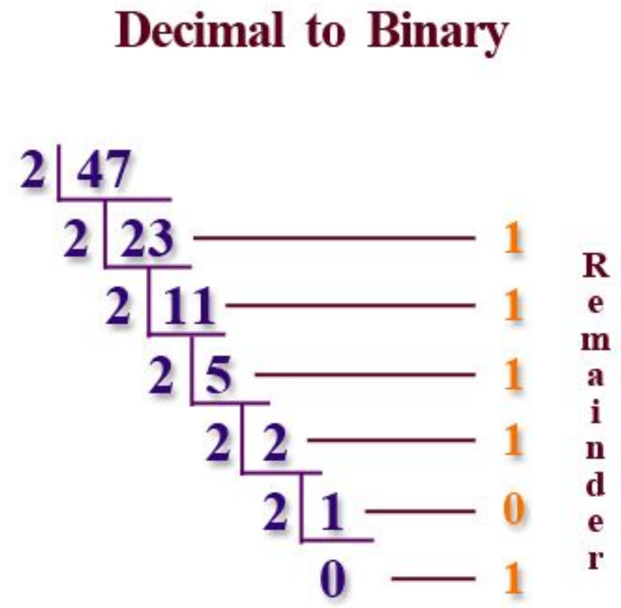
Example:

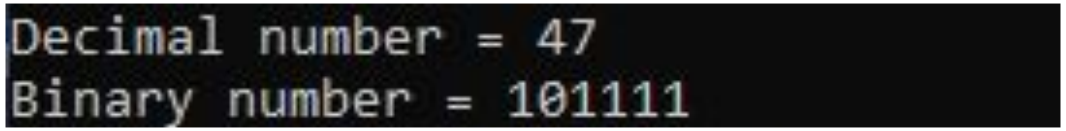
```
include <time.h>

srand(time(NULL));

int rand_num = rand() % n;
```

Exercise 1

- Generate a random number between 0 and 63, and convert it into binary.

  Note that the output should be a decimal integer.



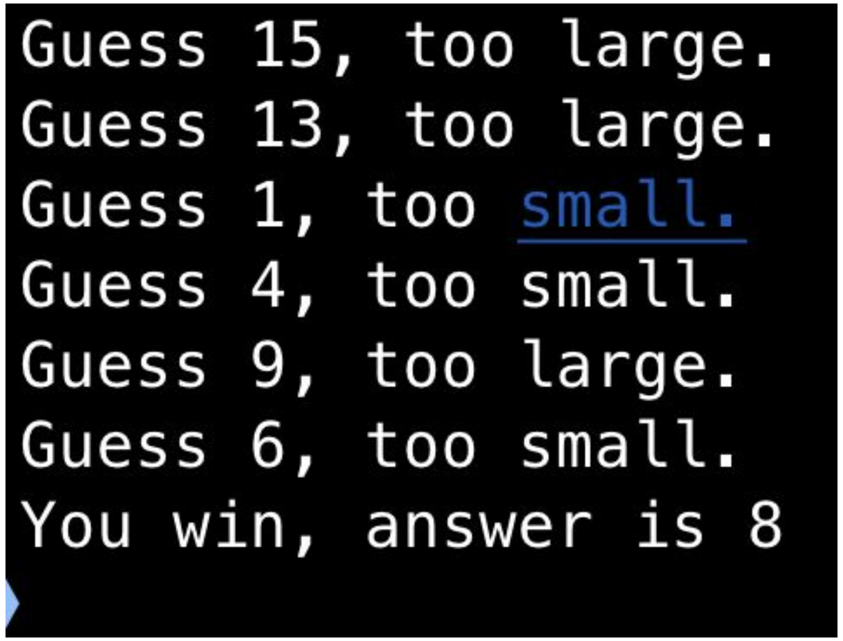


```
Decimal number = 47
Binary number = 101111
```

Exercise 2

Write a program to guess number with range from 0 to 19.

- Set a random answer.
- Generate a random number as guess number in each round.
- Check guess number that is too large or too small.
- Update the random number range.
- Until guess the answer and terminate.



```
Guess 15, too large.
Guess 13, too large.
Guess 1, too small.
Guess 4, too small.
Guess 9, too large.
Guess 6, too small.
You win, answer is 8
```

Exercise Submission Format

Format:

- xxxxxxxxxx_ex_w03.zip
    - xxxxxxxxxx_ex_01.cpp
    - xxxxxxxxxx_ex_02.cpp

xxxxxxxxxx is your student ID