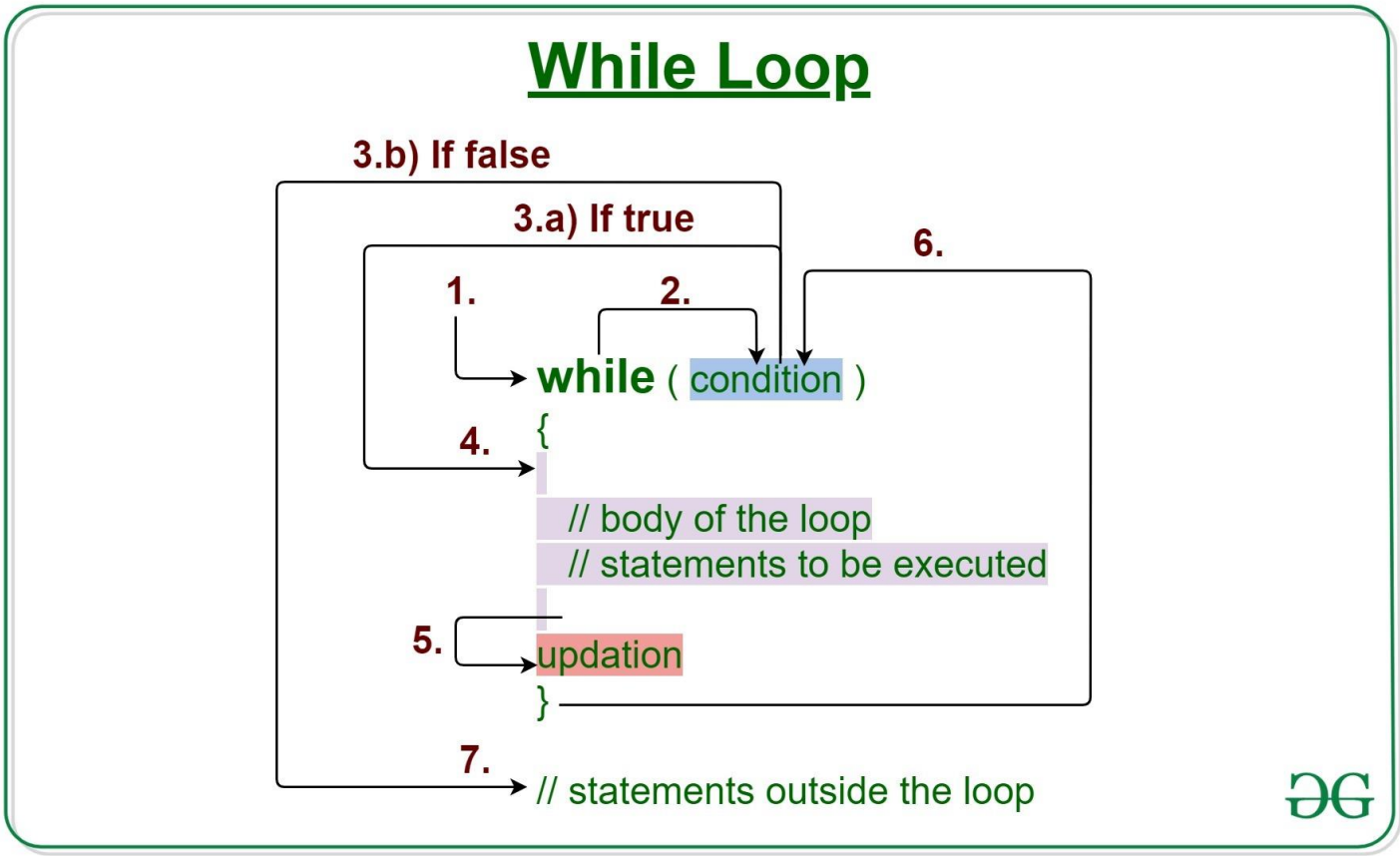
2022-09-25

1. While loop
2. For loop
3. Common errors
4. Exercises

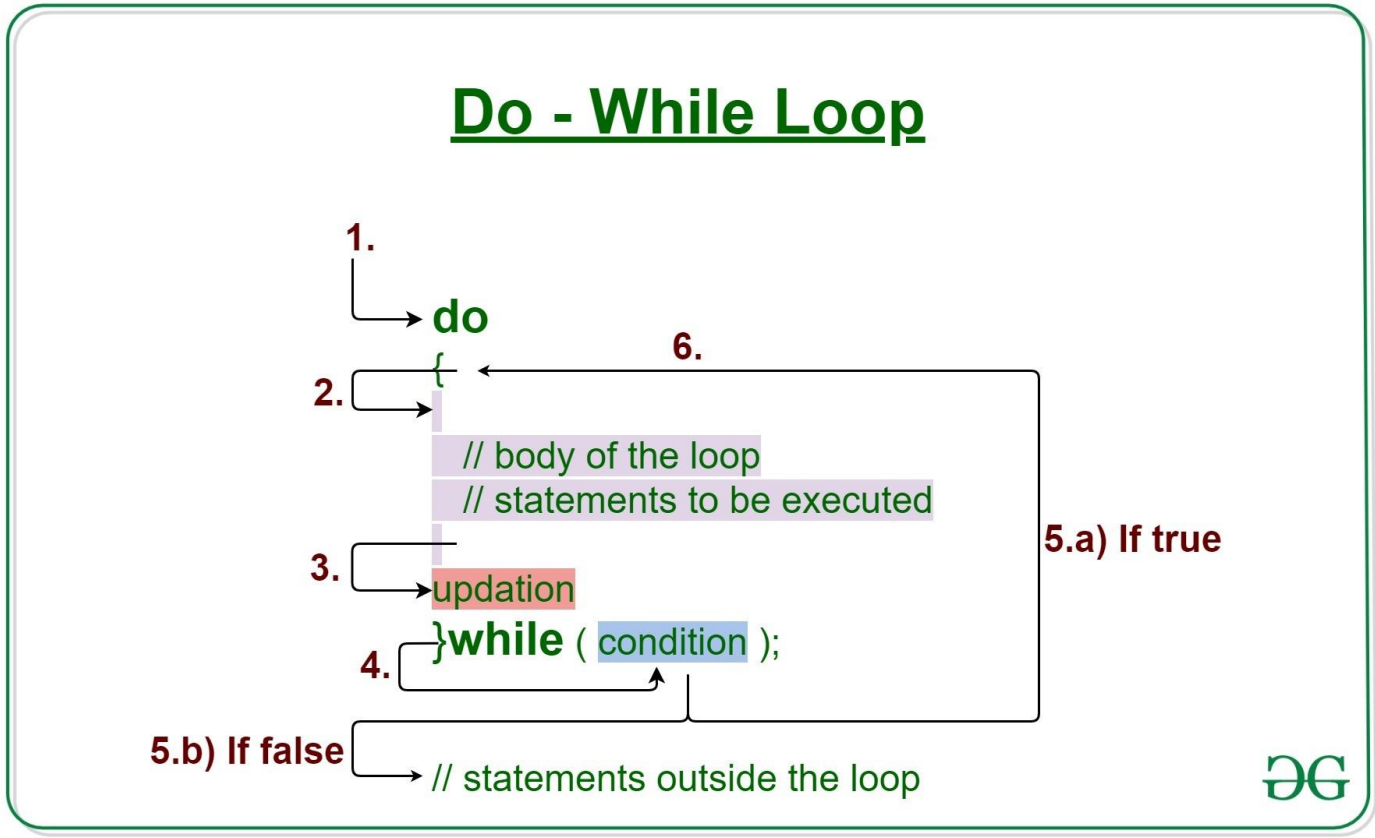
Presenter: Nguyen Khoa

1. WHILE LOOP



https://www.geeksforgeeks.org/java-while-loop-with-examples/

1. WHILE LOOP



https://www.geeksforgeeks.org/java-do-while-loop-with-examples/

1. WHILE LOOP

Source: __

2. FOR LOOP

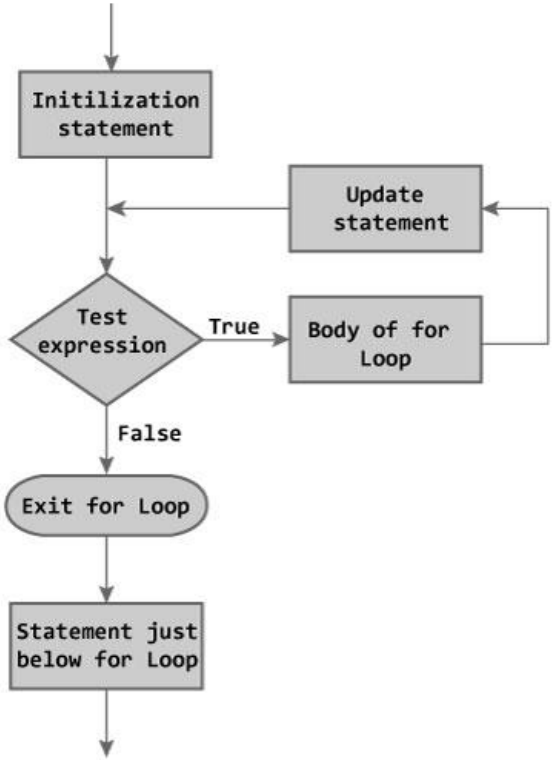
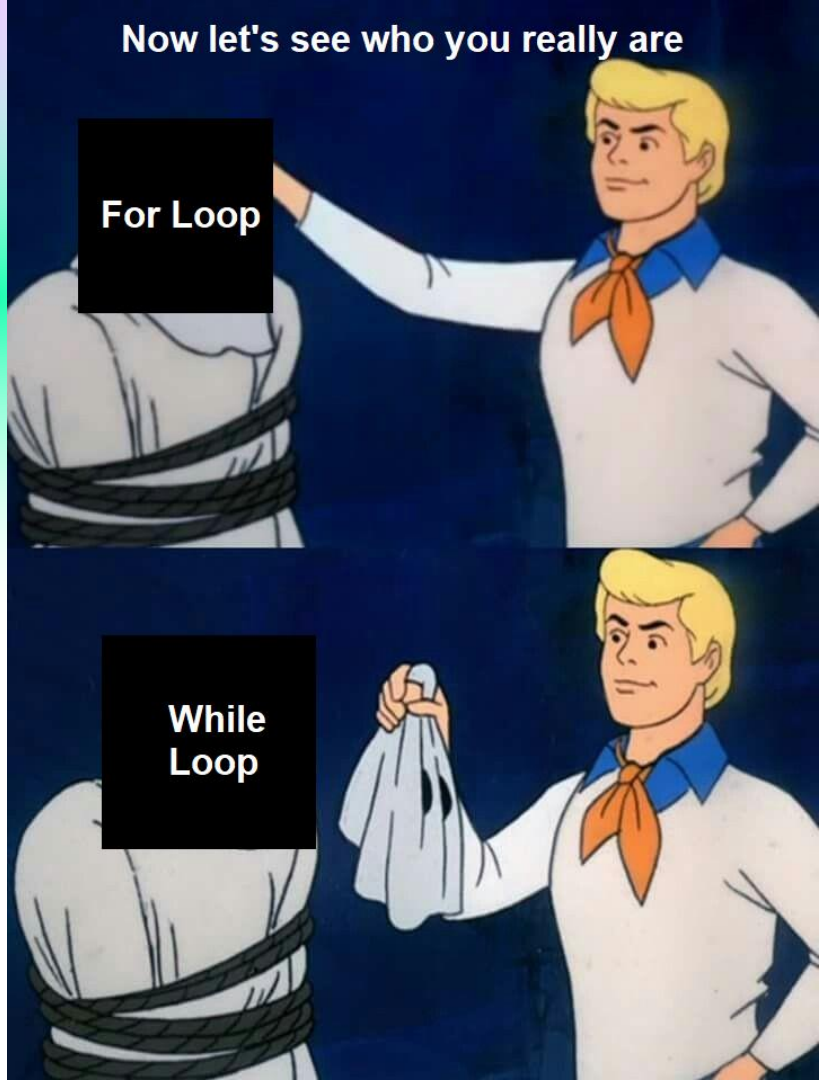


Figure: Flowchart of for Loop

```c
#include <stdio.h>

int main()
{
    // format of for loop
    for (Initilization; Condition; Update)
    {
        // do something
    }
}
```

https://sites.google.com/site/clanguageforlearn/loop-in-c/for-loop



https://www.reddit.com/r/ProgrammerHumor/comments/9tmazc/thrown_for_a_loop/


for (int i = 0; i < 50; ++i) {
// ..
}
Lawful Good
int i = 0;
while (i < 50) {
// ..
++i;
}
Neutral Good
int i;
for (i = 0;i < 50;) {
// ..
++i;
}
Chaotic Good
int i = 0;
do {
// ..
++i;
} while (i < 50);
Lawful Neutral
int i = 0;
while (true) {
if (i == 50) {
break;
}
// ..
++i;
}
True Neutral
for (int i = 0; i <
Double.POSITIVE_INFINITY;) {
if (i == 50) {
break;
}
// ..
++i;
if (i >= 0) {
continue;
}
}
Chaotic Neutral
int i = 0;
loop: for (;;) {
for (;; ++i) {
if (i == 50) {
break loop;
}
// ..
}
}
Lawful Evil
int i = 0;
loop: for (;;) {
if (i < 50) {
// ..
++i;
} else {
break loop;
}
}
Neutral Evil
int i = 0;
loop: while(true != false) {
if(i < 50 + 1) {
if (i == 50) {
break loop;
}
}
// ..
++i;
}
Chaotic Evil

Source: __

3. COMMON ERRORS

```c
// sum of element [0; 5]
int sum = 0;
for (int i=0; i<=5; i++)
{
    sum += i;
    printf("Sum = %d\n", sum);
    i++;
}
```

```
PS D:\01.Code\00.Github\VGU-CA\CSE2021 - Programming 1\2022-09-25\loop_common_error> gcc -o test .\double_increment.c; .\test.exe
Sum = 0
Sum = 2
Sum = 6
```

```c
// solve
int sum = 0;
for (int i=0; i<=5; i++)
{
    sum += i;
    printf("Sum = %d\n", sum);
}
```

```
PS D:\01.Code\00.Github\VGU-CA\CSE2021 - Programming 1\2022-09-25\loop_common_error> gcc -o test .\double_increment.c; .\test.exe
Sum = 0
Sum = 1
Sum = 3
Sum = 6
Sum = 10
Sum = 15
```

Double Increment

3. COMMON ERRORS

Infinite Loop

```c
// calculate the sum from 5 to 10
int i = 5;
int sum = 0;
while (i <= 10){
    sum += i;
    printf("%d\t", sum);
}
```

```
PS D:\01.Code\00.Github\VGU-CA\CSE2021 - Programming 1\2022-09-25\loop_common_error> gcc -o test .\infinite_loop.c; .\test.exe
5       10      15      20      25      30      35      40      45      50      55      60      65      70      75      80      85      90      95      100     105     110     115     120     125     130     135     140     1
45      150     155     160     165     170     175     180     185     190     195     200     205     210     215     220     225     230     235     240     245     250     255     260     265     270     275     280     2
85      290     295     300     305     310     315     320     325     330     335     340     345     350     355     360     365     370     375     380     385     390     395     400     405     410     415     420     4
25      430     435     440     445     450     455     460     465     470     475     480     485     490     495     500     505     510     515     520     525     530     535     540     545     550     555     560     5
65      570     575     580     585     590     595     600     605     610     615     620     625     630     635     640     645     650     655     660     665     670     675     680     685     690     695     700     7
05      710     715     720     725     730     735     740     745     750     755     760     765     770     775     780     785     790     795     800     805     810     815     820     825     830     835     840     8
45      850     855     860     865     870     875     880     885     890     895     900     905     910     915     920     925     930     935     940     945     950     955     960     965     970     975     980     9
85      990     995     1000    1005    1010    1015    1020    1025    1030    1035    1040    1045    1050    1055    1060    1065    1070    1075    1080    1085    1090    1095    1100    1105    1110    1115    1120    1
125     1130    1135    1140    1145    1150    1155    1160    1165    1170    1175    1180    1185    1190    1195    1200    1205    1210    1215    1220    1225    1230    1235    1240    1245    1250    1255    1260    1
265     1270    1275    1280    1285    1290    1295    1300    1305    1310    1315    1320    1325    1330    1335    1340    1345    1350    1355    1360    1365    1370    1375    1380    1385    1390    1395    1400    1
405     1410    1415    1420    1425    1430    1435    1440    1445    1450    1455    1460    1465    1470    1475    1480    1485    1490    1495    1500    1505    1510    1515    1520    1525    1530    1535    1540    1
545     1550    1555    1560    1565    1570    1575    1580    1585    1590    1595    1600    1605    1610    1615    1620    1625    1630    1635    1640    1645    1650    1655    1660    1665    1670    1675    1680    1
685     1690    1695    1700    1705    1710    1715    1720    1725    1730    1735    1740    1745    1750    1755    1760    1765    1770    1775    1780    1785    1790    1795    1800    1805    1810    1815    1820    1
825     1830    1835    1840    1845    1850    1855    1860    1865    1870    1875    1880    1885    1890    1895    1900    1905    1910    1915    1920    1925    1930    1935    1940    1945    1950    1955    1960    1
```

```c
// solve
int i = 5;
int sum = 0;
while (i <= 10){
    sum += i;
    printf("%d\n", sum);
    i++;
}
```

```
PS D:\01.Code\00.Github\VGU-CA\CSE2021 - Programming 1\2022-09-25\loop_common_error> gcc -o test .\infinite_loop.c; .\test.exe
5
11
18
26
35
45
```

3. COMMON ERRORS

```c
// count + print number [0; 5)
int count = 0;
for (int i = 0; i < 5; i++)
    count++;
    printf("me sitting at %d\n", count);
```

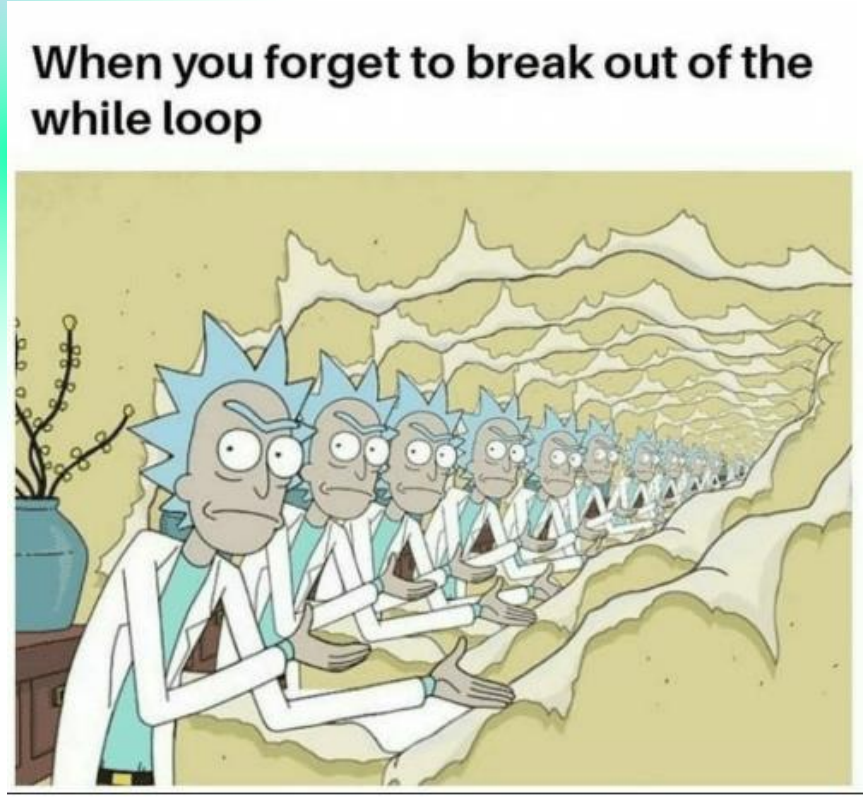
```
PS D:\01.Code\00.Github\VGU-CA\CSE2021 - Programming 1\2022-09-25\loop_common_error> gcc -o test .\not_in_loop.c; .\test.exe
me sitting at 5
```

```c
// solve
int count = 0;
for (int i = 0; i < 5; i++){
    count++;
    printf("me sitting at %d\n", count);
}
```

Not in Loop

```
PS D:\01.Code\00.Github\VGU-CA\CSE2021 - Programming 1\2022-09-25\loop_common_error> gcc -o test .\not_in_loop.c; .\test.exe
me sitting at 1
me sitting at 2
me sitting at 3
me sitting at 4
me sitting at 5
```

3. COMMON ERRORS



The loop

3. COMMON ERRORS

```c
// print the following array
int array[] = {4, 3, 3, 2, 1};
for (int i = 0; i <= 5; i++)
    printf("%d\t", array[i]);
puts("");
```

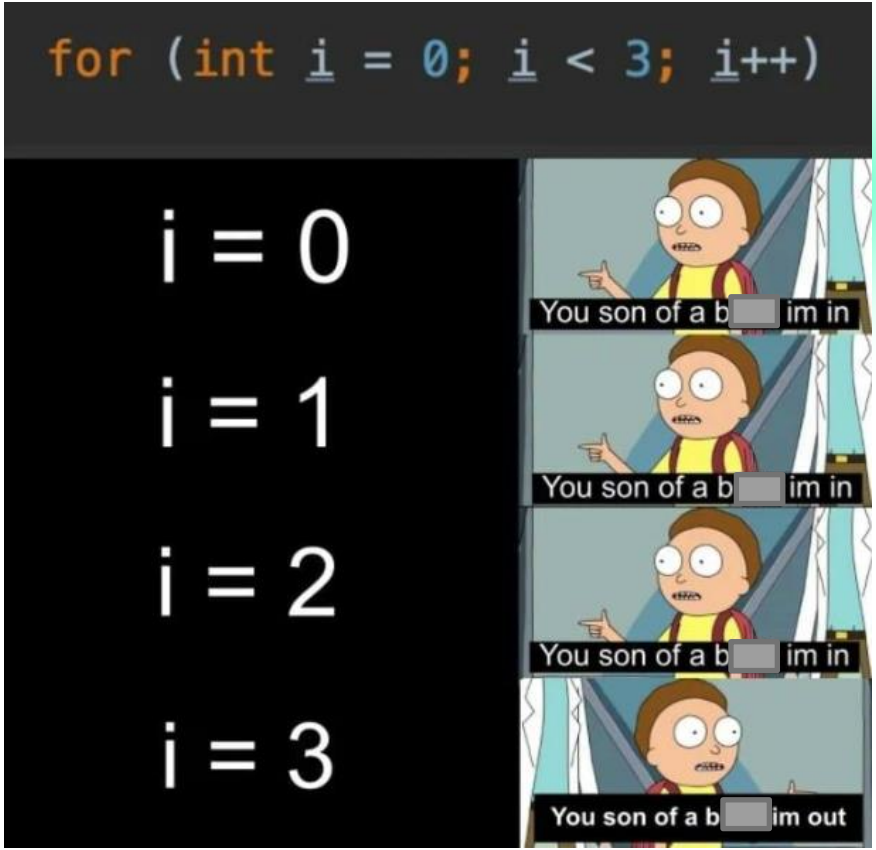
```
PS D:\01.Code\00.Github\VGU-CA\CSE2021 - Programming 1\2022-09-25\loop_common_error> gcc -o test .\off_by_one.c; .\test.exe
4       3       3       2       1       5
```

```c
// solve
int array[] = {4, 3, 3, 2, 1};
for (int i = 0; i < 5; i++)
    printf("%d\t", array[i]);
puts("");
```

Off-by-One

```
PS D:\01.Code\00.Github\VGU-CA\CSE2021 - Programming 1\2022-09-25\loop_common_error> gcc -o test .\off_by_one.c; .\test.exe
4       3       3       2       1
```

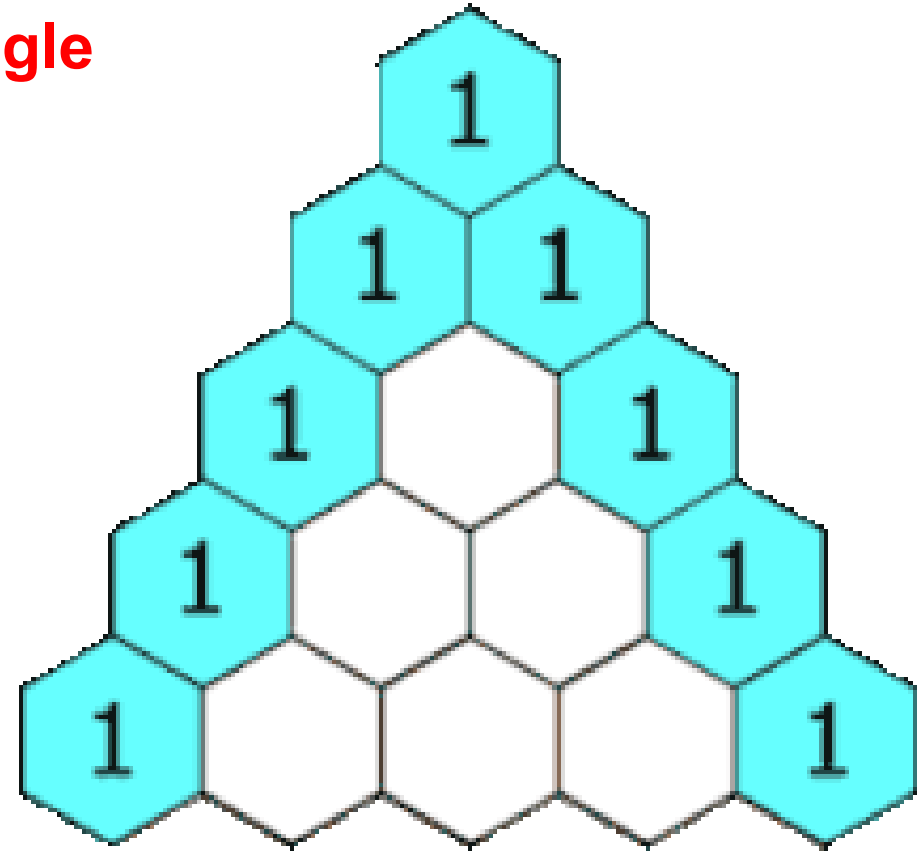
3. COMMON ERRORS



https://www.reddit.com/r/memes/comments/ehls89/understanding_the_for_loop/

4. EXERCISES

Pascal's Triangle



https://en.wikipedia.org/wiki/Pascal%27s_triangle

4. EXERCISES

Pascal's Triangle

```c
/// @file pascal_dynamic.c

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

/**
 * Print tje given array
 * @param[in] arr the array
 * @param[out] _size size of the array
 */
void i_array_print_1d (int *arr, int _size)
{
    for(int i=0; i<_size; i++)
        if (arr[i] != 0)
            printf("%3d ", arr[i]);
    puts("");
}

/**
 * Create a new dynamic array with the given size
 * @param[in] n the size of the array
 * @param[out] _ array with given size
 */
int* i_array_new_1d(int n)
{
    return (int*) calloc(n+1, sizeof(int));
}
```

```c
/**
 * Print the Pascal's Triangle with the given level
 * @param[in] n Pascal's Triangle up to n levels
 */
int pascal_dynamic(int n)
{
    int* result = i_array_new_1d(n+1);
    result[0] = 1;
    for (int i=0; i<=n; i++)
    {
        for (int j=i; j>0; j--)
            result[j] += result[j - 1];
        for(int j = 0; j <= n-i; j++)
            printf("  ");
        i_array_print_1d(result, n+1);
    }
}

/**
 * Main entry point of the program.
*/
int main()
{
    int level;

    printf("Enter the level of Pascal's Triangle:\n>> ");
    scanf("%d", &level);
    pascal_dynamic(level);
}
```

```
Enter the level of Pascal's Triangle:
>> 10
                    1
                  1   1
                1   2   1
              1   3   3   1
            1   4   6   4   1
          1   5  10  10   5   1
        1   6  15  20  15   6   1
      1   7  21  35  35  21   7   1
    1   8  28  56  70  56  28   8   1
  1   9  36  84 126 126  84  36   9   1
1  10  45 120 210 252 210 120  45  10   1
```

Pascal's Triangle

$$\binom{0}{0}$$

$$\binom{1}{0} \quad \binom{1}{1}$$

$$\binom{2}{0} \quad \binom{2}{1} \quad \binom{2}{2}$$

$$\binom{3}{0} \quad \binom{3}{1} \quad \binom{3}{2} \quad \binom{3}{3}$$

$$\binom{4}{0} \quad \binom{4}{1} \quad \binom{4}{2} \quad \binom{4}{3} \quad \binom{4}{4}$$

$$\binom{5}{0} \quad \binom{5}{1} \quad \binom{5}{2} \quad \binom{5}{3} \quad \binom{5}{4} \quad \binom{5}{5}$$

4. EXERCISES

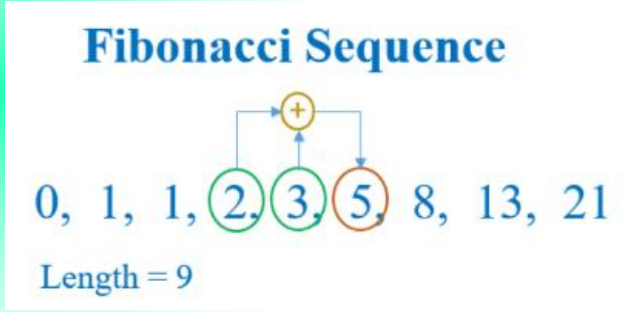
Pascal's Triangle

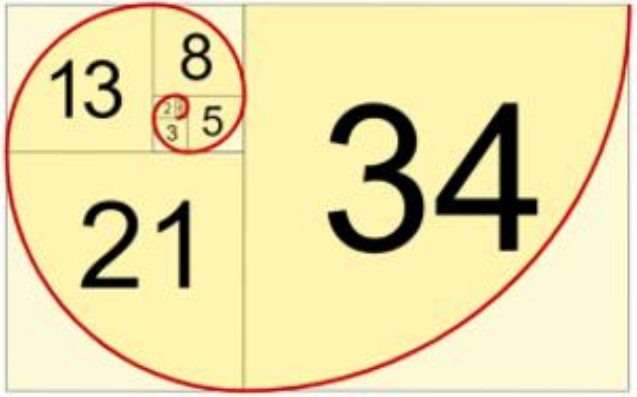
```c
/// @file pascal_algebraic.c

#include <stdio.h>

/**
 * Calculates the factorial
 * Formula: n! = n * (n-1) * (n-2) * ... * 1
 * @param[in] n
 * @param[out] fac
 */
int factorial(int n)
{
    int fac = 1;

    for(; n > 1; n--)
        fac *= n;

    return fac;
}

/**
 * Calculates combination
 * Formula: nCr = n! / ((n-r)! * r!)
 * @param[in] n
 * @param[in] r
 * @param[out] _ combination
 */
int nCr(int n, int r)
{
    return factorial(n) / (factorial(n-r) * factorial(r));
}
```

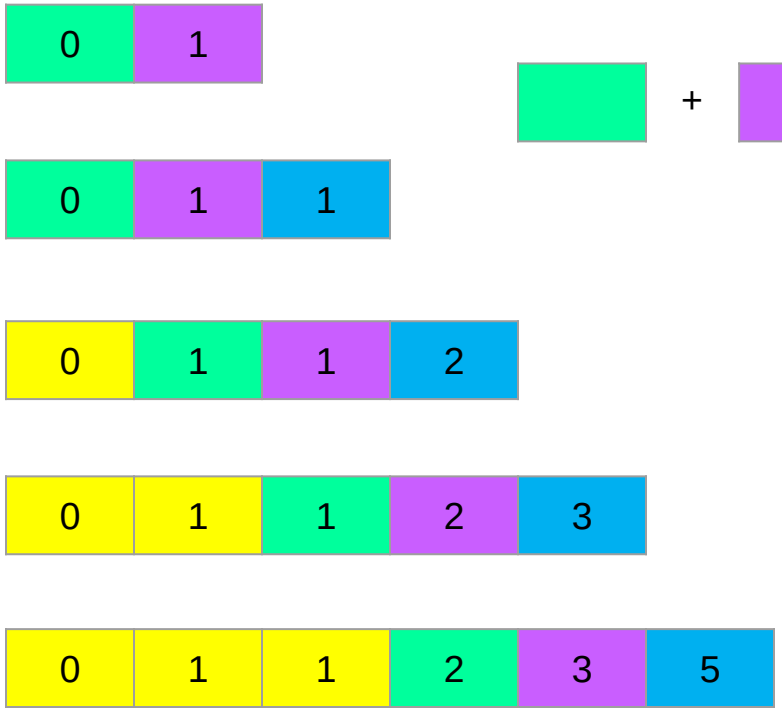
```c
/**
 * Print the Pascal's Triangle with the given level
 * @param[in] n Pascal's Triangle up to n levels
 */
int pascal_algebraic(int n)
{
    for(int i = 0; i <= n; i++)
    {
        for(int j = 0; j <= n-i; j++)
            printf("  ");

        for(int j = 0; j <= i; j++)
            printf(" %3d", nCr(i, j));

        puts("");
    }
}

/**
 * Main entry point of the program.
*/
int main()
{
    int level;

    printf("Enter the level of Pascal's Triangle:\n>> ");
    scanf("%d", &level);
    pascal_algebraic(level);
}
```

```
Enter the level of Pascal's Triangle:
>> 10
                    1
                  1   1
                1   2   1
              1   3   3   1
            1   4   6   4   1
          1   5  10  10   5   1
        1   6  15  20  15   6   1
      1   7  21  35  35  21   7   1
    1   8  28  56  70  56  28   8   1
  1   9  36  84 126 126  84  36   9   1
1  10  45 120 210 252 210 120  45  10   1
```

4. EXERCISES


Fibonacci Sequence
0, 1, 1, 2, 3, 5, 8, 13, 21
Length = 9


13
8
5
21
34


0
1
0
1
1
0
1
1
2
0
1
1
2
3
0
1
1
2
3
5


+
=

4. EXERCISES

Fibonacci Sequence

```c
/// @file fibonacci.c

#include <stdio.h>

/**
 * Print the Fibonacci sequence
 * @param[in] length the length of Fibonacci sequence
 */
void fibonacci(int length)
{
    int i;
    int first = 0;
    int second = 1;
    int next;

    printf("\nThe Fibonacci series:\n");
    for ( i = 0 ; i < length ; i++ )
    {
        if ( i <= 1 )
            next = i;
        else
            next = first + second;
            first = second;
            second = next;
        printf("%d\t",next);
    }
}
```

```c
/**
 * Main entry point of the program.
*/
int main()
{
    int length;

    printf("Enter the length of sequence:\n>> ");
    scanf("%d", &length);
    fibonacci(length);
}
```

```
Enter the length of sequence:
>> 9
The Fibonacci series:
0       1       1       2       3       5       8       13      21
```

4. EXERCISES

```c
/**
 * Calculates the sine of a number
 * Formula (Latex): sin(x) = (-1)^{n}\sum_{n=0}^{\infty}\frac{x^{(2n+1)}}
 * @param[in] x number you wanna calculate sine
 * @param[in] n number of loops
 * @param[out] result sine of x
 */
double sine(double x, int n)
{
    double result = 0;
    for (int i = 0; i <= n; i++)
        result += pow(-1, i) * pow(x, 2 * i + 1) / factorial(2 * i + 1);
    return result;
}

/**
 * Main entry point of the program.
*/
int main()
{
    double num;
    int loop;

    printf("Enter the number for sine: \n>> ");
    scanf("%lf", &num);
    printf("Enter the number of loop: \n>> ");
    scanf("%d", &loop);
    printf("sin(x=%f, loop=%d) = %f\n", num, loop, sine(num, loop));
}
```

$$sin(x) = (-1)^n \sum_{n=0}^{\infty} \frac{x^{(2n+1)}}{(2n+1)!} = x - \frac{x^3}{3!} + \frac{x^5}{5!} - \frac{x^7}{7!} + \frac{x^9}{9!} - \dots$$

```c
/// @file sine.c

#include <stdio.h>
#include <math.h>

/**
 * Calculates the factorial
 * Formula: n! = n * (n-1) * (n-2) * ... * 1
 * @param[in] n
 * @param[out] fac
 */
double factorial(int n)
{
    double fac = 1;

    for(; n > 1; n--)
        fac *= n;

    return fac;
}
```

```
Enter the number for sine:
>> 3.14
Enter the number of loop:
>> 10
sin(x=3.140000, loop=10) = 0.001593
```

4. EXERCISES

```c
/// @file cosine.c

#include <stdio.h>
#include <math.h>

/**
 * Calculates the factorial
 * Formula: n! = n * (n-1) * (n-2) * ... * 1
 * @param[in] n
 * @param[out] fac
 */
double factorial(int n)
{
    double fac = 1;

    for(; n > 1; n--)
        fac *= n;

    return fac;
}

/**
 * Calculates the cosine of a number
 * Formula (Latex): cos(x) = (-1)^{n}\sum_{n=0}^{\infty}\frac{x^{
 * @param[in] x number you wanna calculate cosine
 * @param[in] n number of loops
 * @param[out] result cosine of x
 */
double cosine(double x, int n)
{
    double result = 0;
    for (int i = 0; i <= n; i++)
        result += pow(-1, i) * pow(x, 2 * i) / factorial(2 * i);
    return result;
}
```

$$cos(x) = (-1)^n \sum_{n=0}^{\infty} \frac{x^{2n}}{(2n)!} = 1 - \frac{x^2}{2!} + \frac{x^4}{4!} - \frac{x^6}{6!} + \frac{x^8}{8!} - \dots$$

```c
/**
 * Main entry point of the program.
*/
int main()
{
    double num;
    int loop;

    printf("Enter the number for cosine: \n>> ");
    scanf("%lf", &num);
    printf("Enter the number of loop: \n>> ");
    scanf("%d", &loop);
    printf("cos(x=%f, loop=%d) = %f\n", num, loop, cosine(num, loop));
}
```

```
Enter the number for cosine:
>> 3.14
Enter the number of loop:
>> 10
sin(x=3.140000, loop=10) = -0.999999
```

4. EXERCISES

```c
/// @file sinh.c

#include <stdio.h>
#include <math.h>

/**
 * Calculates the factorial
 * Formula: n! = n * (n-1) * (n-2) * ... * 1
 * @param[in] n
 * @param[out] fac
 */
double factorial(int n)
{
    double fac = 1;

    for (int i = 1; i <= n; i++)
        fac *= i;

    return fac;
}
```

$$sinh(x) = \sum_{n=0}^{\infty} \frac{x^{2n}}{(2n)!} = x + \frac{x^3}{3!} + \frac{x^5}{5!} + \frac{x^7}{7!} + \frac{x^9}{9!} + \dots$$

```c
/**
 * Calculates the sine of a number
 * Formula (Latex): sinh(x) = \sum_{n=0}^{\infty}\frac{x^{2n}}{(2n)!} = x + \frac
 * @param[in] x number you wanna calculate sine
 * @param[in] n number of loops
 * @param[out] result sine of x
 */
double hyperbolic_sine(double x, int n)
{
    double sum;

    for (int i = 0; i < n; i++)
        sum += pow(x, 2*i+1) / factorial(2*i+1);

    return sum;
}

/**
 * Main entry point of the program.
*/
int main()
{
    double num;
    int loop;

    printf("Enter the number for hyperbolic sine: \n>> ");
    scanf("%lf", &num);
    printf("Enter the number of loop: \n>> ");
    scanf("%d", &loop);
    printf("sinh(x=%f, loop=%d) = %f\n", num, loop, hyperbolic_sine(num, loop));
}
```

```
Enter the number for hyperbolic sine:
>> 3.14
Enter the number of loop:
>> 10
sinh(x=3.140000, loop=10) = 11.530292
```

4. EXERCISES

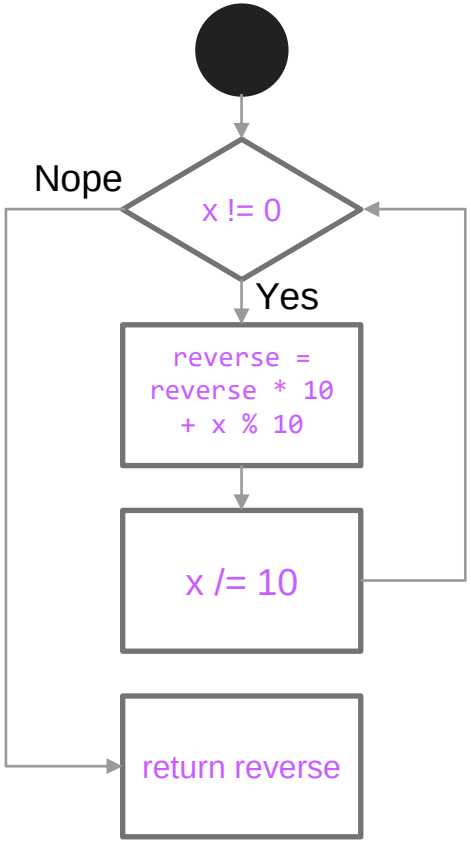
```c
/// @file cosh.c

#include <stdio.h>
#include <math.h>

/**
 * Calculates the factorial
 * Formula: n! = n * (n-1) * (n-2) * ... * 1
 * @param[in] n
 * @param[out] fac
 */
double factorial(int n)
{
    double fac = 1;

    for (int i = 1; i <= n; i++)
        fac *= i;

    return fac;
}
```

```c
/**
 * Calculates the hyperbolic cosine using for loop
 * Formula (Latex): cosh(x) = \sum_{n=0}^{\infty}\frac{x^{2n}}{(2n)!} = 1 + \frac{x^
 * @param[in] x the number want to calculate
 * @param[in] n the number of loops
 * @param[out] sum hyperbolic cosine of x with n loops
 */
double hyperbolic_cosine(double x, int n)
{
    double sum;

    for (int i = 0; i < n; i++)
        sum += pow(x, 2*i) / factorial(2*i);

    return sum;
}

/**
 * Main entry point of the program.
 */
int main()
{
    double num;
    int loop;

    printf("Enter the number for hyperbolic cosine: \n>> ");
    scanf("%lf", &num);
    printf("Enter the number of loop: \n>> ");
    scanf("%d", &loop);
    printf("cosh(x=%f, loop=%d) = %f\n", num, loop, hyperbolic_cosine(num, loop));
}
```

```
Enter the number for hyperbolic cosine:
>> 3.14
Enter the number of loop:
>> 10
cosh(x=3.140000, loop=10) = 11.573575
```

$$cosh(x) = \sum_{n=0}^{\infty} \frac{x^{2n}}{(2n)!} = 1 + \frac{x^2}{2!} + \frac{x^4}{4!} + \frac{x^6}{6!} + \frac{x^8}{8!} + \dots$$

Reverse a Number



4. EXERCISES

```c
/// @file reverse_number_while.c

#include <stdio.h>

/**
 * Reverse given number but it's while loop
 * Example: 12345678910 => (0)1987654321
 * @param[in] x number that u wanna reverse
 * @param[out] reverse reversed number
 */
long long int reverse_number(long long int x)
{
    int reverse = 0; ///< variable to store the result

    while (x != 0)
    {
        reverse = reverse * 10 + x % 10;
        x /= 10;
    }
    return reverse;
}

/**
 * Main entry point of the program.
*/
int main()
{
    long long int n = 12345678910;
    printf("Number = %lld\n", n);
    long long int result = reverse_number(n);
    printf("Reversed number = %lld", result);
}
```

Reverse a Number

```
Number = 12345678910
Reversed number = 1987654321
```

4. EXERCISES

```c
/// @file reverse_number_for.c

#include <stdio.h>

/**
 * Reverse given number but it's for loop
 * Example: 12345678910 => (0)1987654321
 * @param[in] x number that u wanna reverse
 * @param[out] reverse reversed number
 */
long long int reverse_number(long long int x)
{
    int reverse = 0; ///< variable to store the result

    for(;x!=0;)
    {
        reverse = reverse * 10 + x % 10;
        x /= 10;
    }

    return reverse;
}

/**
 * Main entry point of the program.
*/
int main()
{
    long long int n = 12345678910;
    printf("Number = %lld\n", n);
    long long int result = reverse_number(n);
    printf("Reversed number = %lld", result);
}
```

Reverse a Number

```
Number = 12345678910
Reversed number = 1987654321
```

4. EXERCISES

```c
/// @file reverse_number_recursive.c

#include <stdio.h>

/**
 * Reverse given number but it's recursive
 * Example: 12345678910 => (0)1987654321
 * @param[in] x number that u wanna reverse
 * @param[in] reverse current reversed number
 * @param[out] reverse final reversed number
 */
long long int reverse_number(long long int x, long long int reverse)
{
    if(x!=0)
        reverse_number(x / 10, reverse * 10 + x % 10);
    else
        return reverse;
}

/**
 * Main entry point of the program.
*/
int main()
{
    long long int n = 12345678910;
    printf("Number = %lld\n", n);
    long long int result = reverse_number(n, 0);
    printf("Reversed number = %lld", result);
}
```

Reverse a Number

```
Number = 12345678910
Reversed number = 1987654321
```