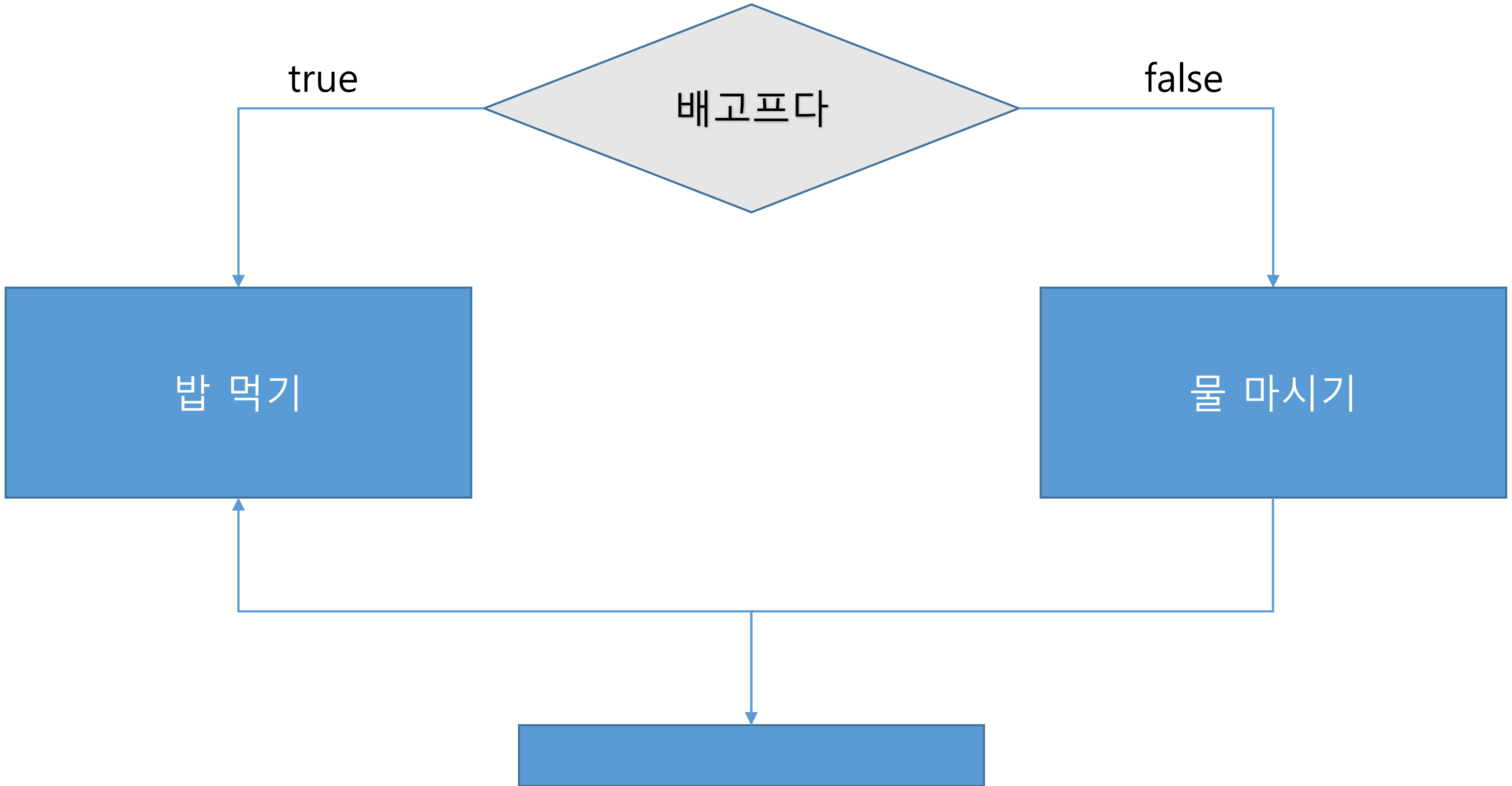
Conditional Statements

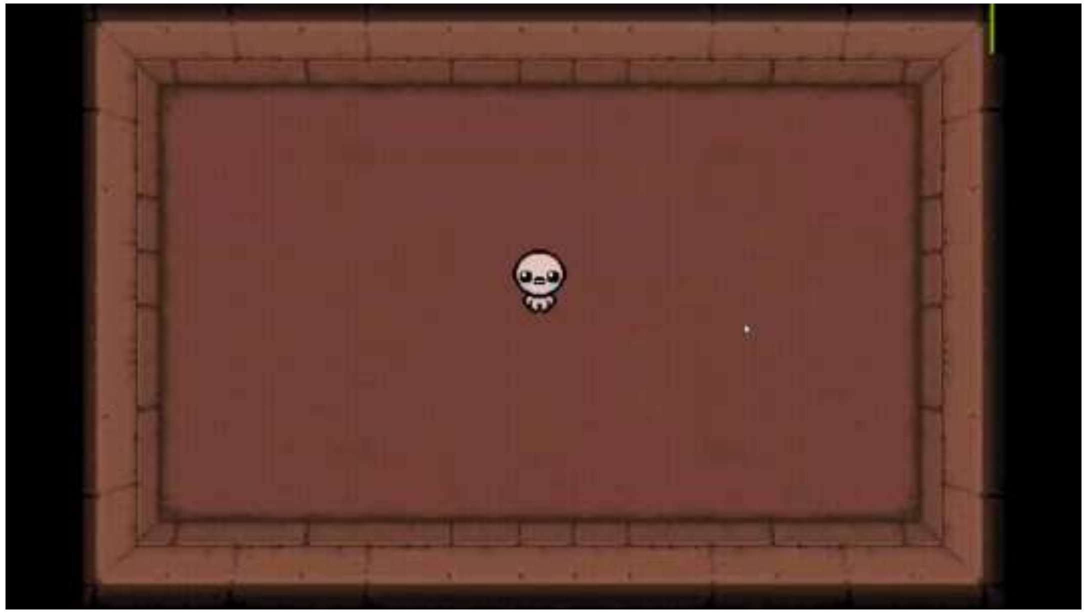
DIGIPEN DRAGONS

Jinwoo Choi

Conditional Statements(조건문) 이란?



Conditional Statements(조건문) 이란?



if in C

```
if(condition)
{
    //statements to execute if condition is true
}
```

if statement executes a block of code when its condition evaluates to true(non-zero)

And skips it when the condition is false(zero)

else in C

```
if(condition)
{
    //statements to execute if condition is true
}
else
{
    //statements to execute if condition is false
}
```

else in C

```
if(condition)
{
    if(condition_1)
    {
        // if condition is true and condition_1 is also true.
    }
    else
    {
        // if condition is true but condition_1 is false.
    }
}
else
{
}
```

else if in C

```
if(condition1)
{
}
else if(condition2)
{
    //statements to execute if condition1 is false and condition2 is true
}
else
{
}
```

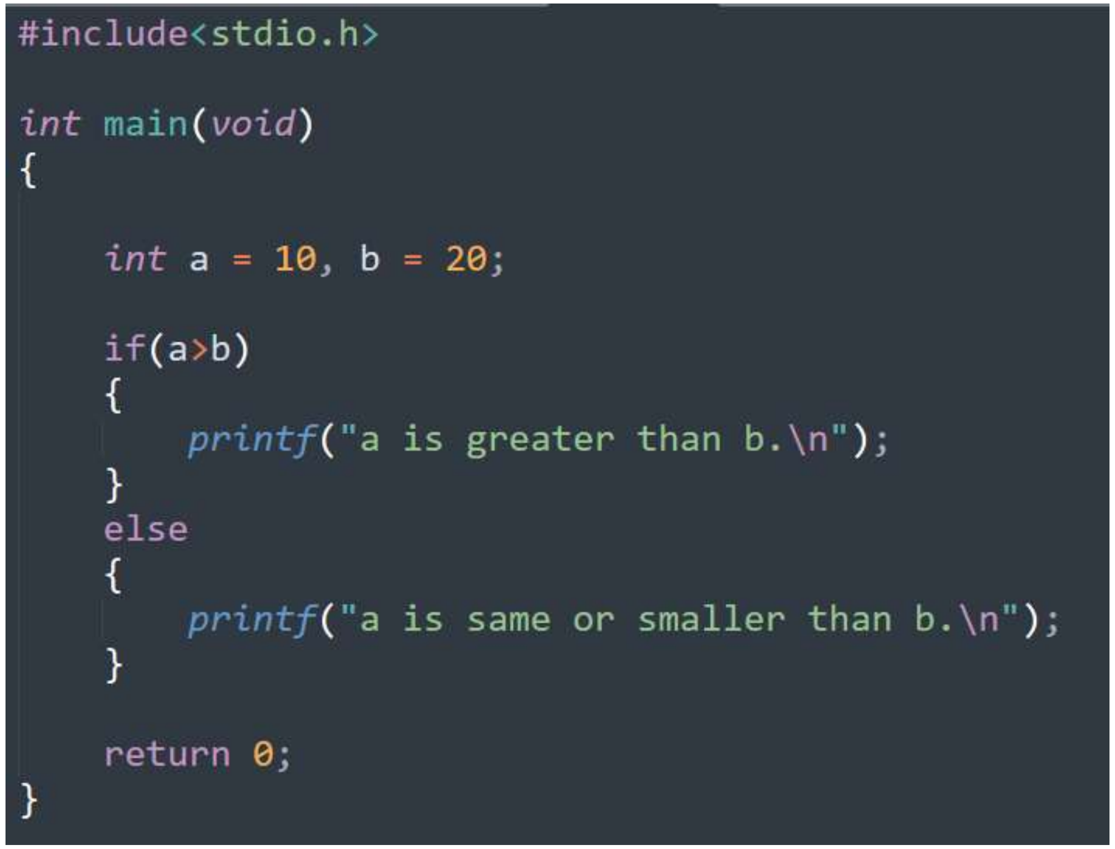
else if in C

```
if(condition1)
{
}
else if(condition2)
{
    //statements to execute if condition1 is false and condition2 is true
}
else if(condition3)
{
    //statements to execute if condition1 and condition2 are false, and condition 3 is true
}
else
{
}
```

Condition Types

Comparison Operators(비교 연산자)

Operator	Description
==	Equal to
!=	Not equal to
>	Greater than
<	Less than
>=	Greater than or equal to
<=	Less than or equal to

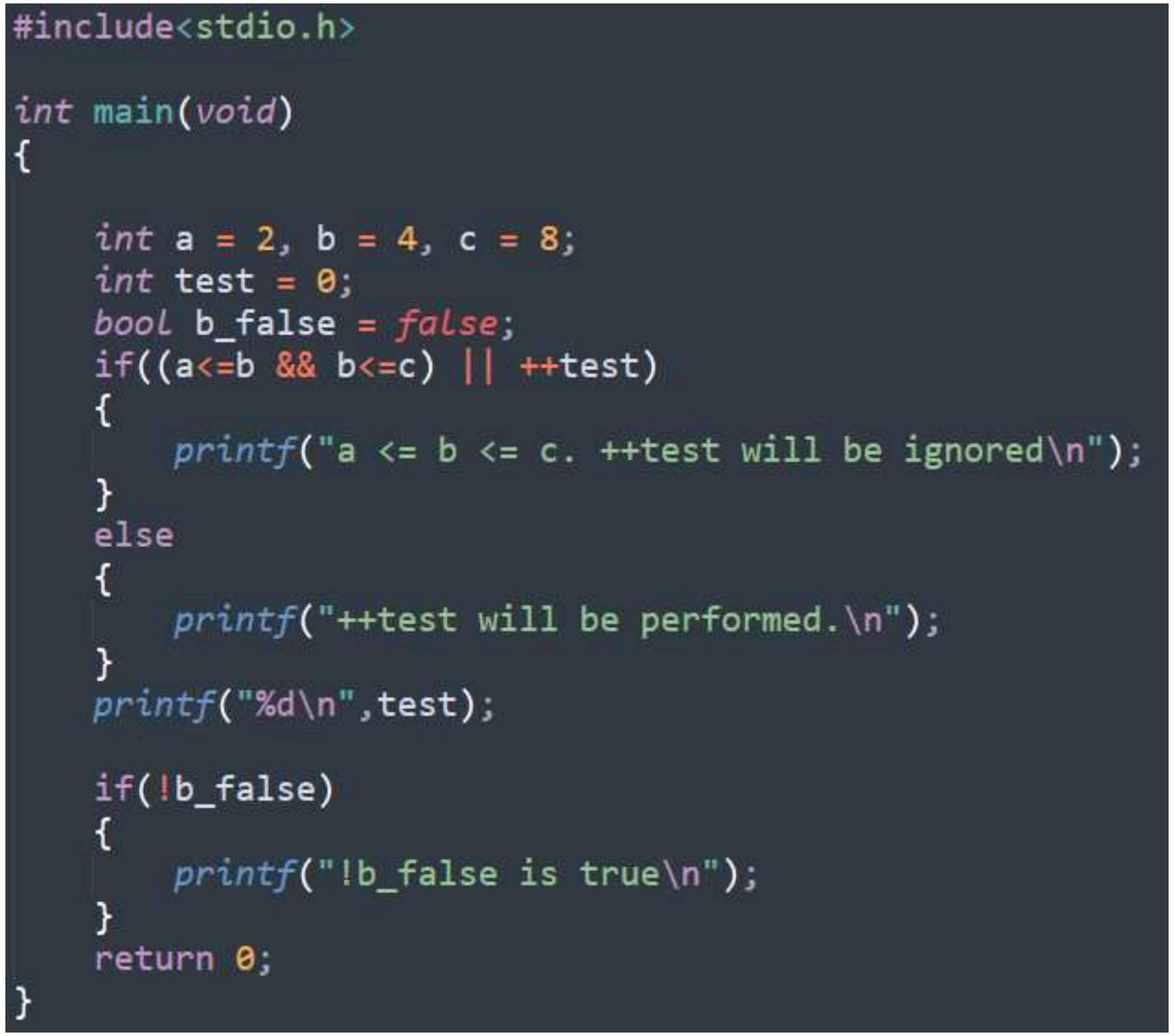


```
#include<stdio.h>

int main(void)
{

    int a = 10, b = 20;

    if(a>b)
    {
        printf("a is greater than b.\n");
    }
    else
    {
        printf("a is same or smaller than b.\n");
    }

    return 0;
}
```

Condition Types

Logical Operators(논리 연산자)

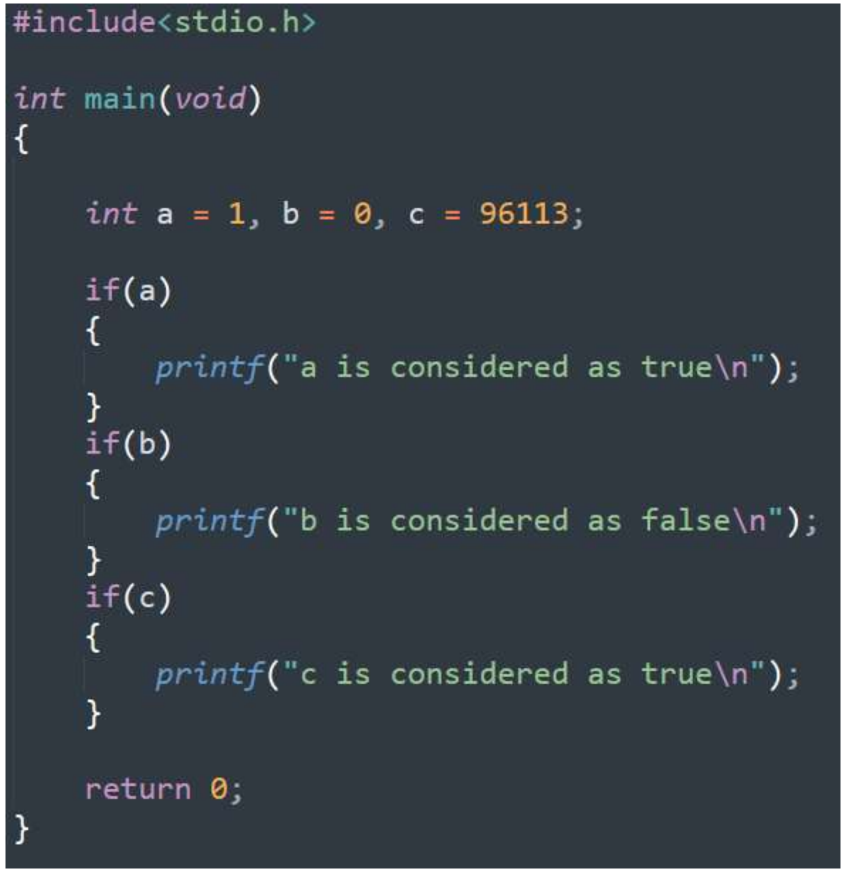
Operator	Description
&&	Logical AND(true if both are true)
||	Logical OR(true if at least one is true)
!	Logical NOT(negates a value)



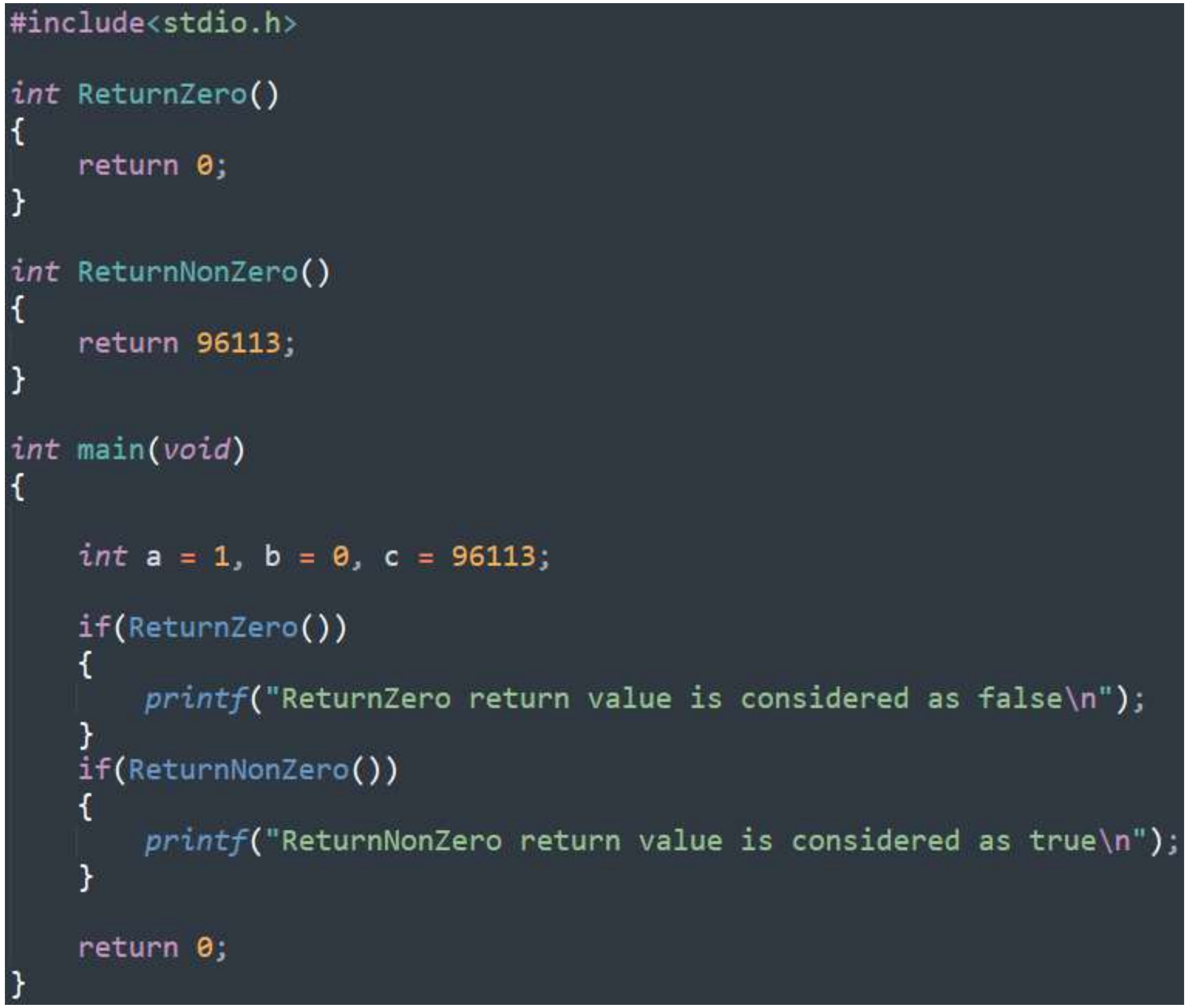
```
#include<stdio.h>

int main(void)
{

    int a = 2, b = 4, c = 8;
    int test = 0;
    bool b_false = false;
    if((a<=b && b<=c) || ++test)
    {
        printf("a <= b <= c. ++test will be ignored\n");
    }
    else
    {
        printf("++test will be performed.\n");
    }
    printf("%d\n",test);

    if(!b_false)
    {
        printf("!b_false is true\n");
    }
    return 0;
}
```

Condition Types

Using Variable or Function Return Values



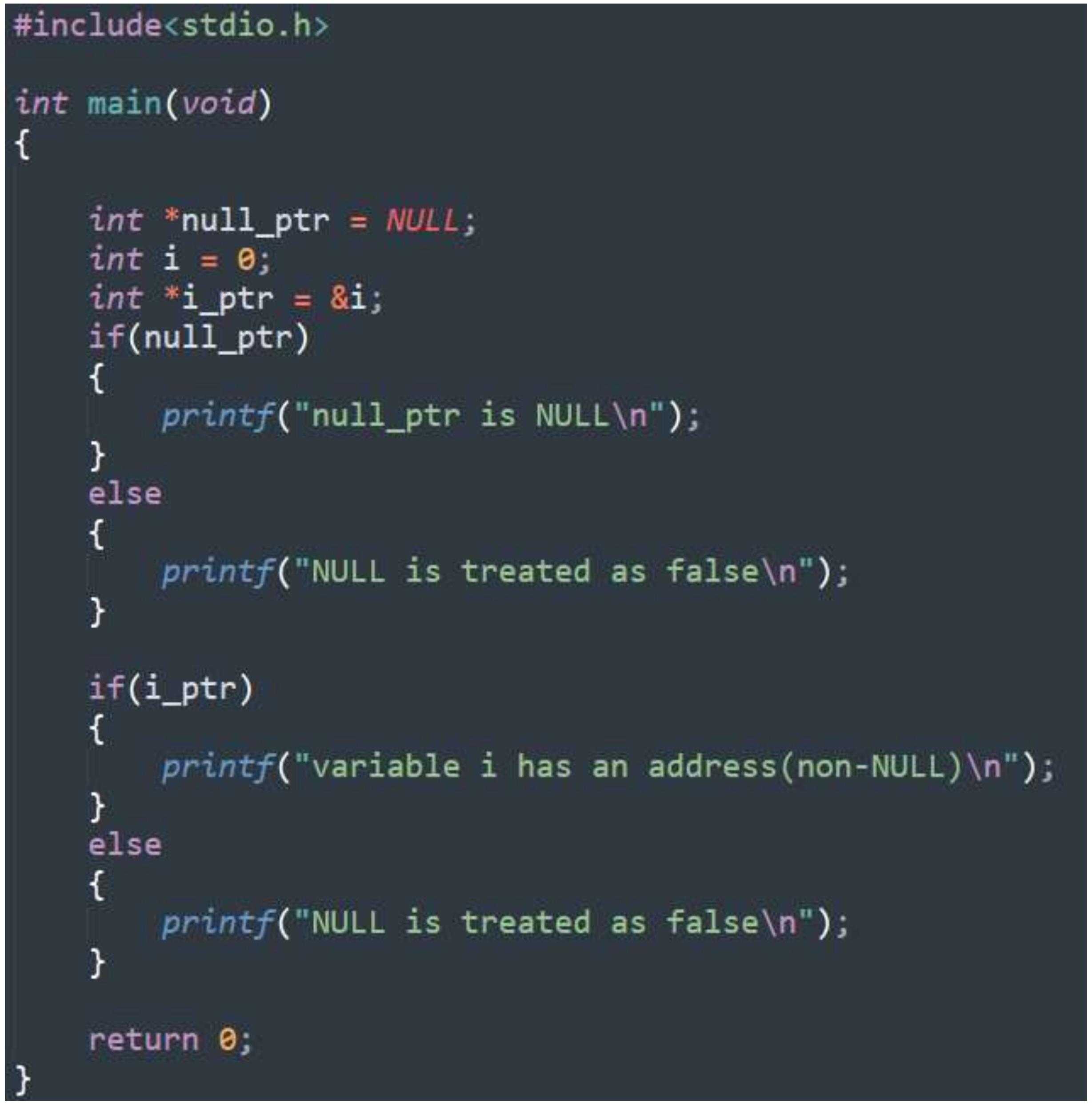
```c
#include<stdio.h>

int main(void)
{

    int a = 1, b = 0, c = 96113;

    if(a)
    {
        printf("a is considered as true\n");
    }
    if(b)
    {
        printf("b is considered as false\n");
    }
    if(c)
    {
        printf("c is considered as true\n");
    }

    return 0;
}
```



```c
#include<stdio.h>

int ReturnZero()
{
    return 0;
}

int ReturnNonZero()
{
    return 96113;
}

int main(void)
{

    int a = 1, b = 0, c = 96113;

    if(ReturnZero())
    {
        printf("ReturnZero return value is considered as false\n");
    }
    if(ReturnNonZero())
    {
        printf("ReturnNonZero return value is considered as true\n");
    }

    return 0;
}
```

Condition Types

Pointer(NULL or non-NULL)

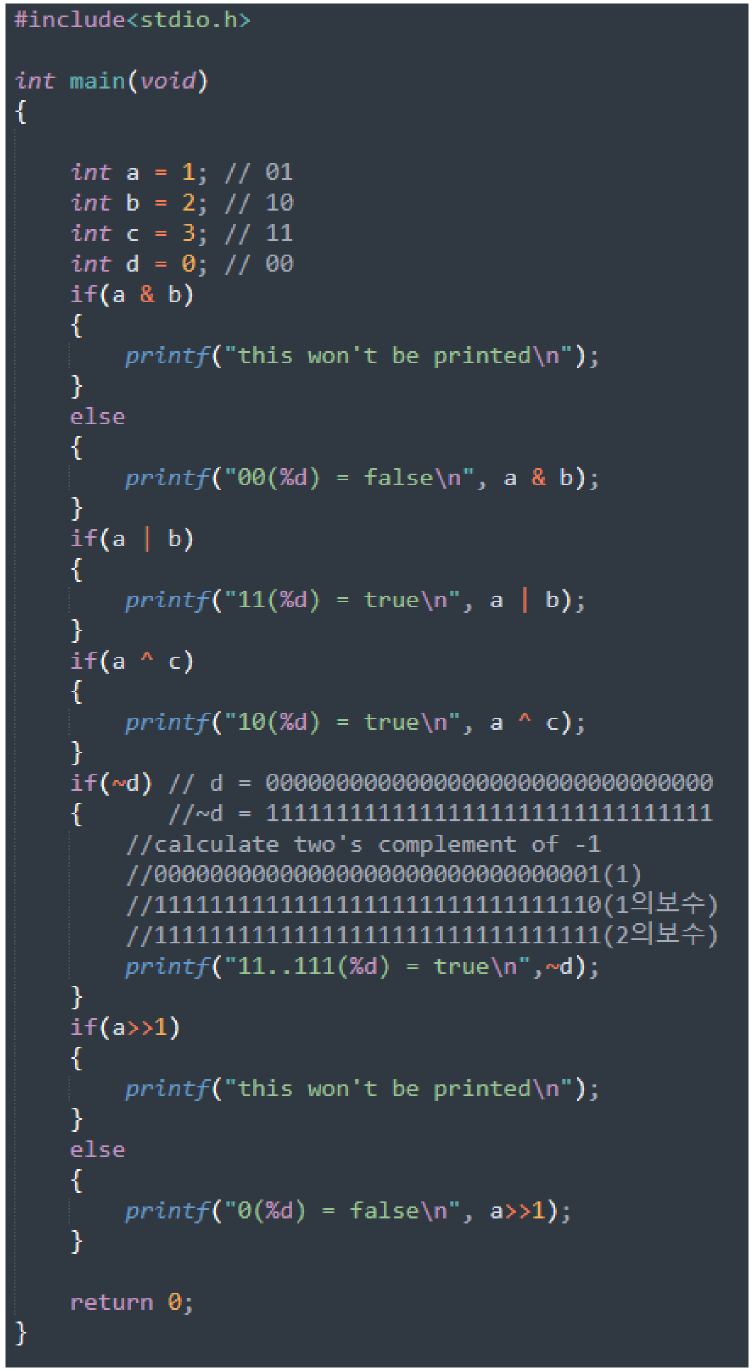


```c
#include<stdio.h>

int main(void)
{

    int *null_ptr = NULL;
    int i = 0;
    int *i_ptr = &i;
    if(null_ptr)
    {
        printf("null_ptr is NULL\n");
    }
    else
    {
        printf("NULL is treated as false\n");
    }

    if(i_ptr)
    {
        printf("variable i has an address(non-NULL)\n");
    }
    else
    {
        printf("NULL is treated as false\n");
    }

    return 0;
}
```

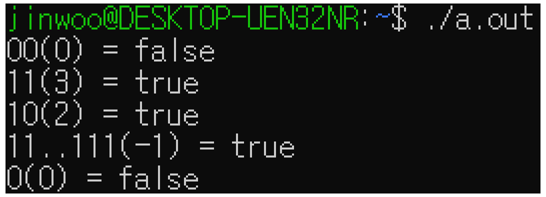
Condition Types

Bitwise Operations

Operator	Description
&	Bitwise AND
|	Bitwise OR
^	Bitwise XOR
~	Bitwise NOT
<<	Left shift
>>	Right shift



```c
#include<stdio.h>

int main(void)
{

    int a = 1; // 01
    int b = 2; // 10
    int c = 3; // 11
    int d = 0; // 00
    if(a & b)
    {
        printf("this won't be printed\n");
    }
    else
    {
        printf("00(%d) = false\n", a & b);
    }
    if(a | b)
    {
        printf("11(%d) = true\n", a | b);
    }
    if(a ^ c)
    {
        printf("10(%d) = true\n", a ^ c);
    }
    if(~d) // d = 00000000000000000000000000000000
    {      //~d = 11111111111111111111111111111111
        //calculate two's complement of -1
        //00000000000000000000000000000001(1)
        //11111111111111111111111111111110(1의보수)
        //11111111111111111111111111111111(2의보수)
        printf("11..111(%d) = true\n",~d);
    }
    if(a>>1)
    {
        printf("this won't be printed\n");
    }
    else
    {
        printf("0(%d) = false\n", a>>1);
    }

    return 0;
}
```

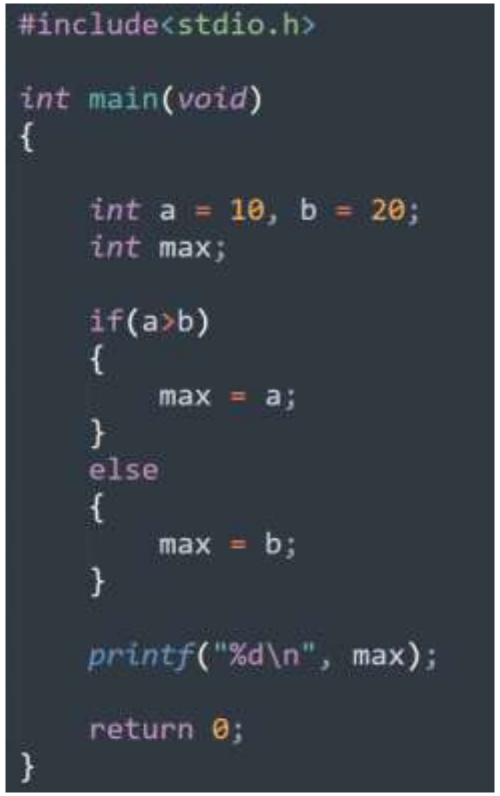


```
jinwoo@DESKTOP-UEN32NR:~$ ./a.out
00(0) = false
11(3) = true
10(2) = true
11..111(-1) = true
0(0) = false
```

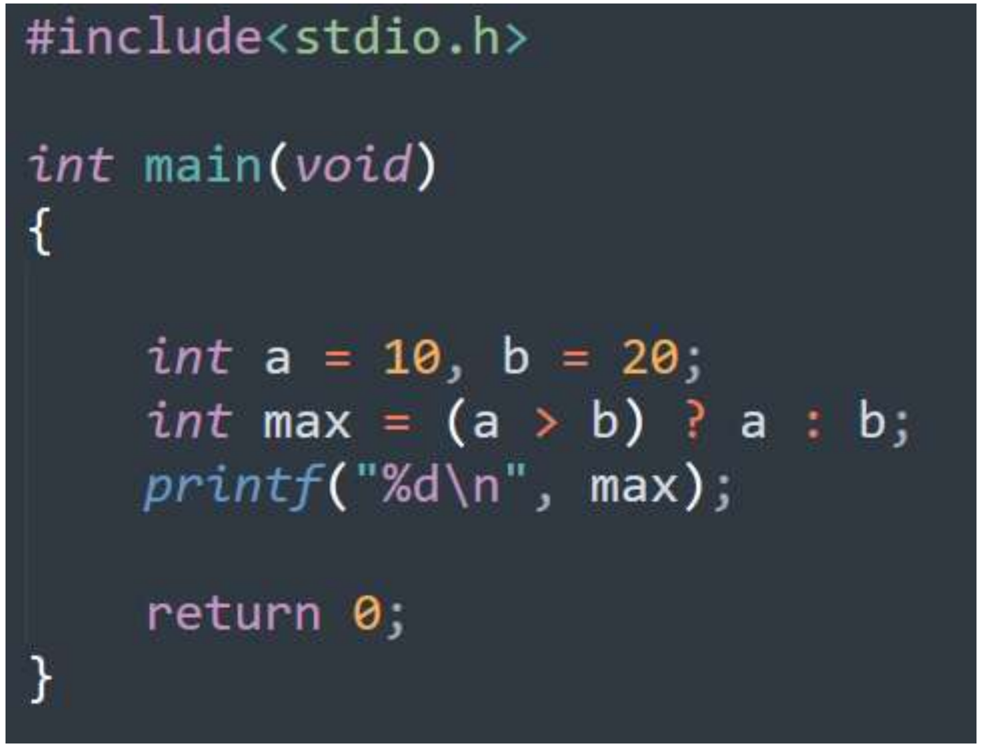
Condition Types

Ternary Operator (? :)

Syntax	Description
(condition) ? true_result : false_result;	A shorthand ‘if-else’ statement



```c
#include<stdio.h>

int main(void)
{

    int a = 10, b = 20;
    int max;

    if(a>b)
    {
        max = a;
    }
    else
    {
        max = b;
    }

    printf("%d\n", max);

    return 0;
}
```



```c
#include<stdio.h>

int main(void)
{

    int a = 10, b = 20;
    int max = (a > b) ? a : b;
    printf("%d\n", max);

    return 0;
}
```

switch-case in C

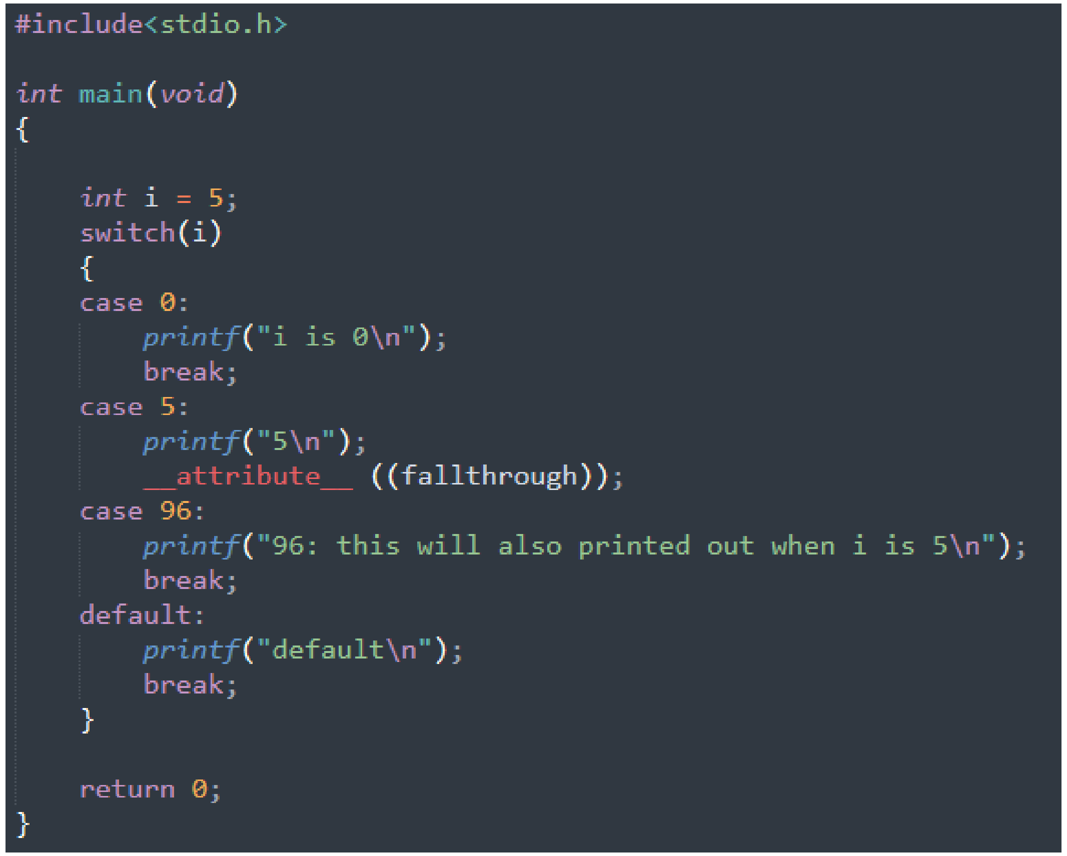
```
switch (variable)
{
case 0:
    //statements to execute if variable value is 0
case 1:
    //statements to execute if variable value is 1
case 2:
    //statements to execute if variable value is 2
case 96113:
    //statements to execute if variable value is 96113
default:
    //statements to execute if all other cases fail
}
```

```
switch (variable)
{
case 0:
    break;
case 1:
case 2:
case 96113:
    break;
default:
    //statements to execute if all other cases fail
    break;
}
```

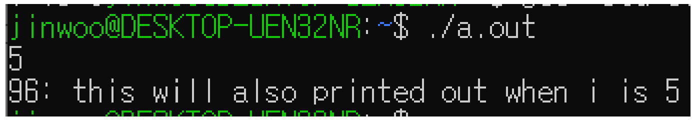
Fall through

switch-case in C

switch-case only works with only **int, char, enum**

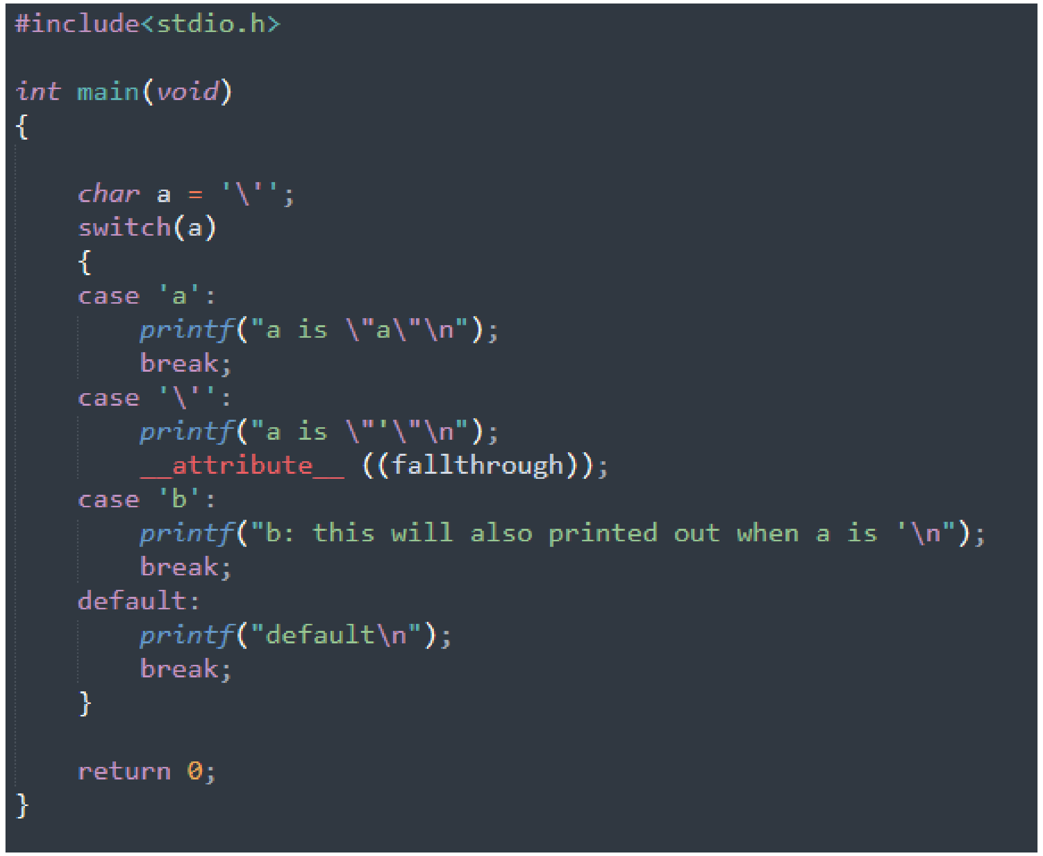


```c
#include<stdio.h>

int main(void)
{

    int i = 5;
    switch(i)
    {
    case 0:
        printf("i is 0\n");
        break;
    case 5:
        printf("5\n");
        __attribute__ ((fallthrough));
    case 96:
        printf("96: this will also printed out when i is 5\n");
        break;
    default:
        printf("default\n");
        break;
    }

    return 0;
}
```

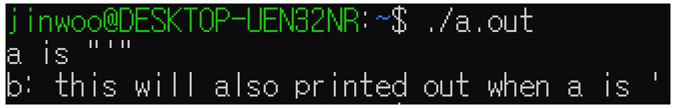


```
jinwoo@DESKTOP-UEN32NR:~$ ./a.out
5
96: this will also printed out when i is 5
```

Faster than if-else for large cases(optimized by jump tables)



```c
#include<stdio.h>

int main(void)
{

    char a = '\'';
    switch(a)
    {
    case 'a':
        printf("a is \"a\"\n");
        break;
    case '\'':
        printf("a is \"'\"\n");
        __attribute__ ((fallthrough));
    case 'b':
        printf("b: this will also printed out when a is '\n");
        break;
    default:
        printf("default\n");
        break;
    }

    return 0;
}
```



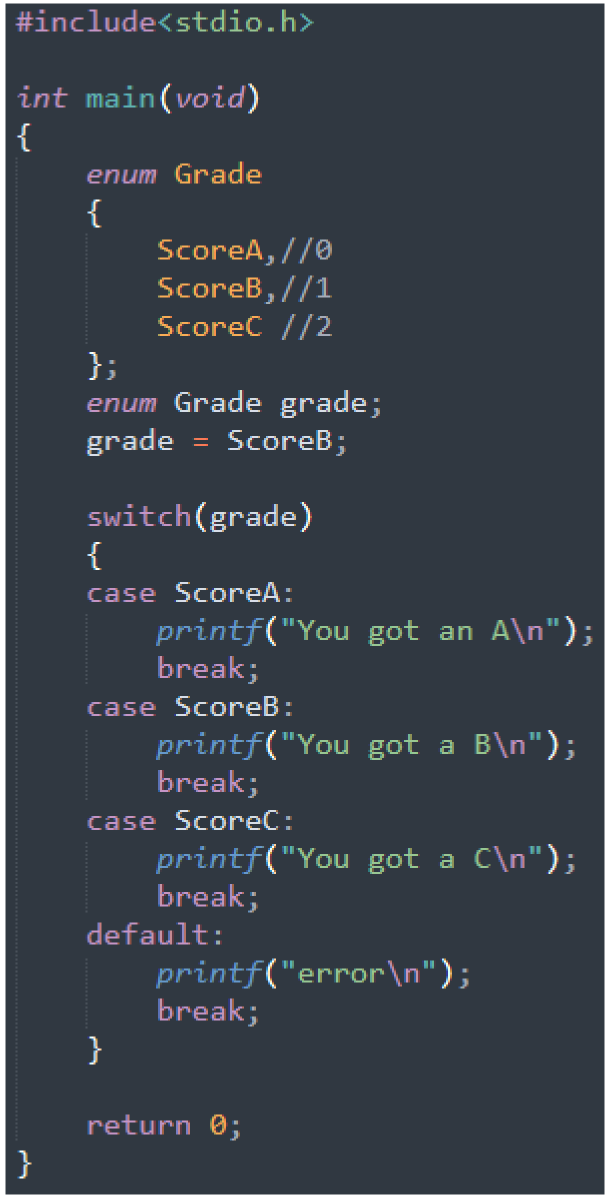
```
jinwoo@DESKTOP-UEN32NR:~$ ./a.out
a is "'"
b: this will also printed out when a is '
```

enum

```
enum enumName
{
    userDefinedName0,
    userDefinedName1,
    userDefinedName2
            .
            .
            .
};
```

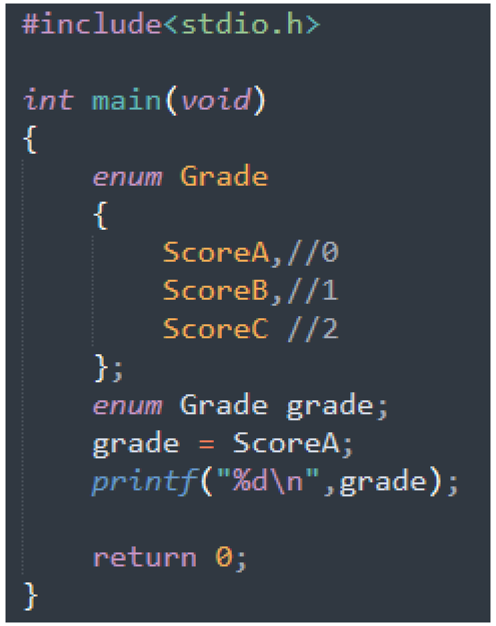
```
enum enumName
{
    userDefinedName0 = 5,
    userDefinedName1 = 12,
    userDefinedName2 = 60
            .
            .
            .
};
```

```
enum enumName variableName;
variableName = userDefinedName0;
```

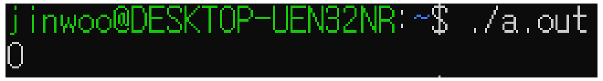


```
#include<stdio.h>

int main(void)
{
    enum Grade
    {
        ScoreA,//0
        ScoreB,//1
        ScoreC //2
    };
    enum Grade grade;
    grade = ScoreB;

    switch(grade)
    {
    case ScoreA:
        printf("You got an A\n");
        break;
    case ScoreB:
        printf("You got a B\n");
        break;
    case ScoreC:
        printf("You got a C\n");
        break;
    default:
        printf("error\n");
        break;
    }

    return 0;
}
```

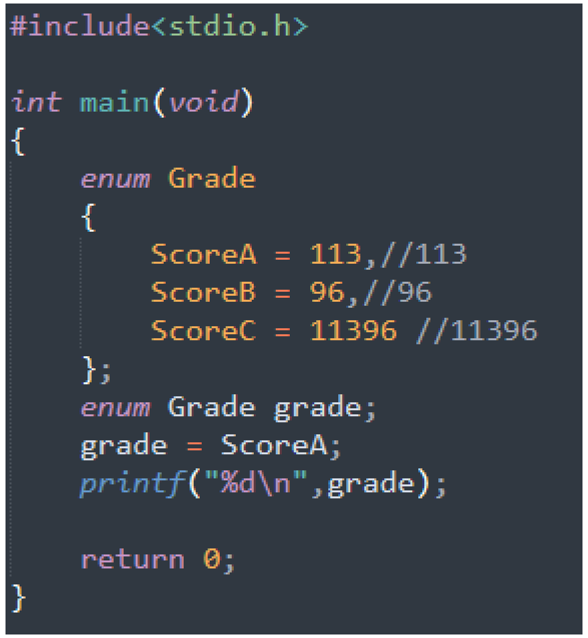
enum



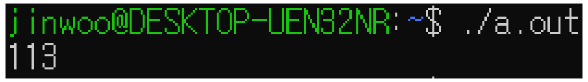
```c
#include<stdio.h>

int main(void)
{
	enum Grade
	{
		ScoreA,//0
		ScoreB,//1
		ScoreC //2
	};
	enum Grade grade;
	grade = ScoreA;
	printf("%d\n",grade);

	return 0;
}
```



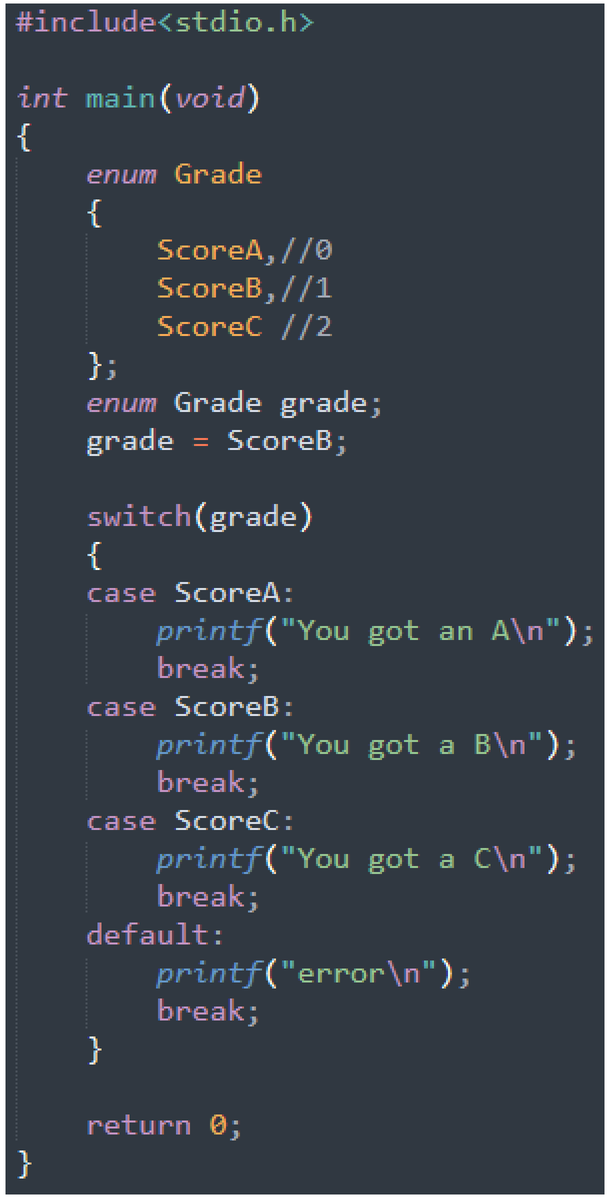
```
jinwoo@DESKTOP-UEN32NR:~$ ./a.out
0
```



```c
#include<stdio.h>

int main(void)
{
	enum Grade
	{
		ScoreA = 113,//113
		ScoreB = 96,//96
		ScoreC = 11396 //11396
	};
	enum Grade grade;
	grade = ScoreA;
	printf("%d\n",grade);

	return 0;
}
```



```
jinwoo@DESKTOP-UEN32NR:~$ ./a.out
113
```



```c
#include<stdio.h>

int main(void)
{
	enum Grade
	{
		ScoreA,//0
		ScoreB,//1
		ScoreC //2
	};
	enum Grade grade;
	grade = ScoreB;

	switch(grade)
	{
	case ScoreA:
		printf("You got an A\n");
		break;
	case ScoreB:
		printf("You got a B\n");
		break;
	case ScoreC:
		printf("You got a C\n");
		break;
	default:
		printf("error\n");
		break;
	}

	return 0;
}
```

LAB – Conditional Statements

- Create a file named 'ConditionalStatements_YourName.c'.
- Your program should get an integer input from the user.
- Use an **if-else statement** to check if it is **positive or negative**.
- Use **switch statement** to check if it is **even or odd**.
- You should print out the result to the console.