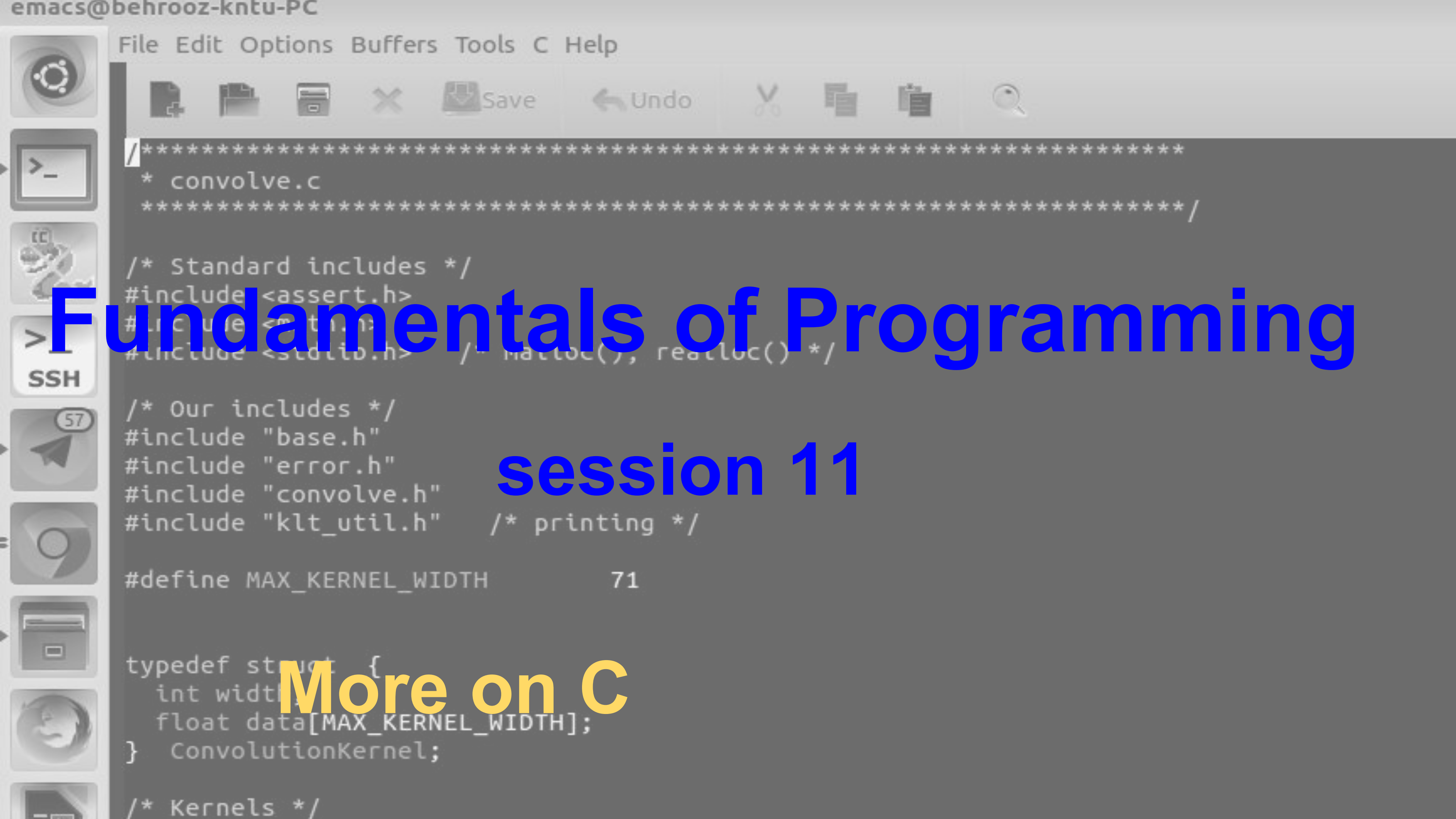
Fundamentals of Programming

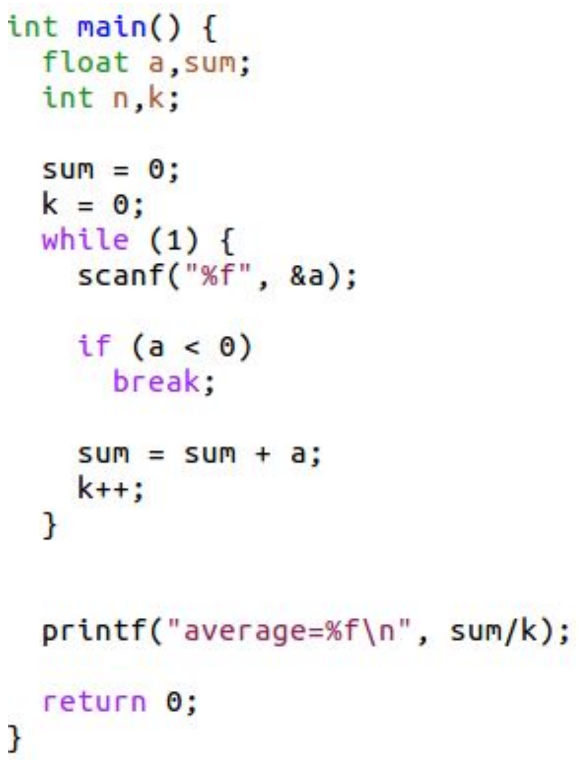
session 11

More on C



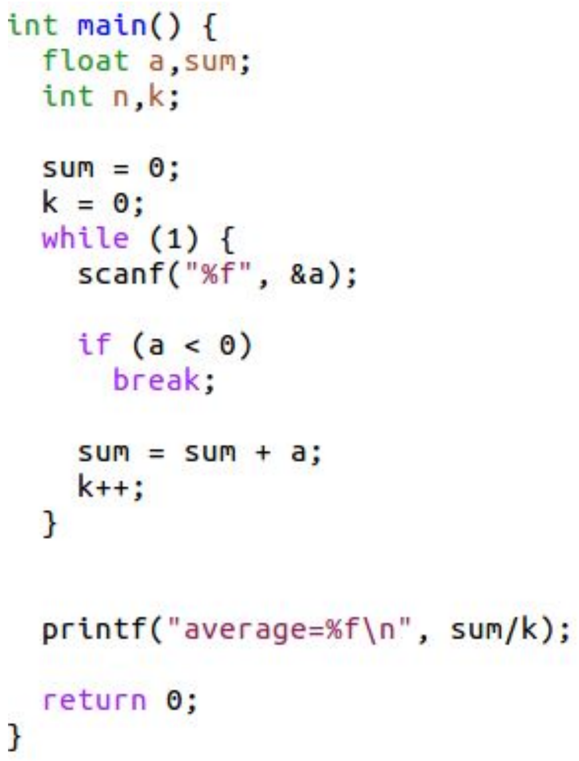
```
/**********************************************************************
 * convolve.c
 *********************************************************************/

/* Standard includes */
#include <assert.h>
#include <math.h>
#include <stdlib.h>   /* malloc(), realloc() */

/* Our includes */
#include "base.h"
#include "error.h"
#include "convolve.h"
#include "klt_util.h"   /* printing */

#define MAX_KERNEL_WIDTH        71


typedef struct  {
  int width;
  float data[MAX_KERNEL_WIDTH];
}  ConvolutionKernel;

/* Kernels */
```

Remember from last session (Average score)



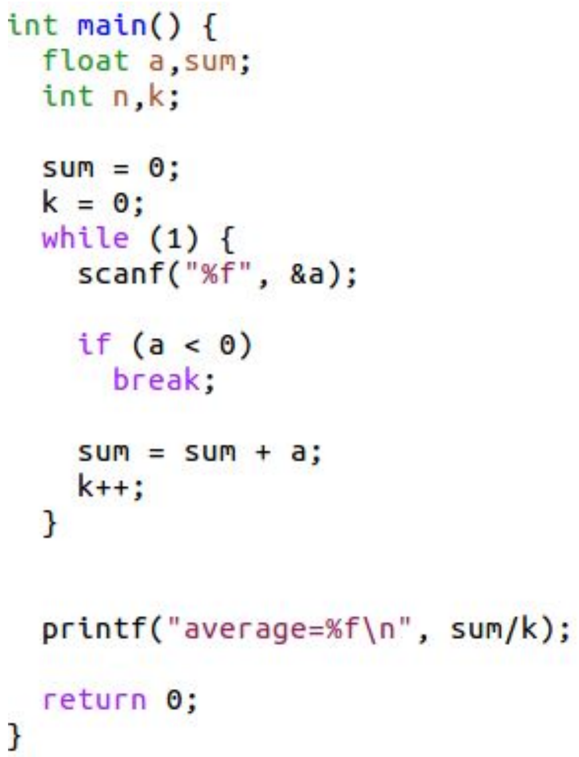
```
int main() {
  float a,sum;
  int n,k;

  sum = 0;
  k = 0;
  while (1) {
    scanf("%f", &a);

    if (a < 0)
      break;

    sum = sum + a;
    k++;
  }


  printf("average=%f\n", sum/k);

  return 0;
}
```

Divide by zero!

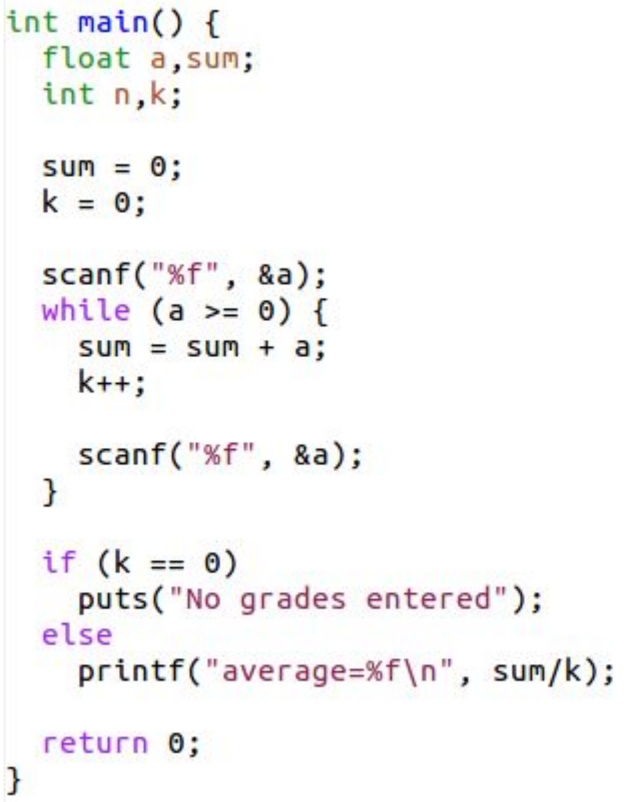


```
int main() {
  float a,sum;
  int n,k;

  sum = 0;
  k = 0;
  while (1) {
    scanf("%f", &a);

    if (a < 0)
      break;

    sum = sum + a;
    k++;
  }


  printf("average=%f\n", sum/k);

  return 0;
}
```

Divide by zero!

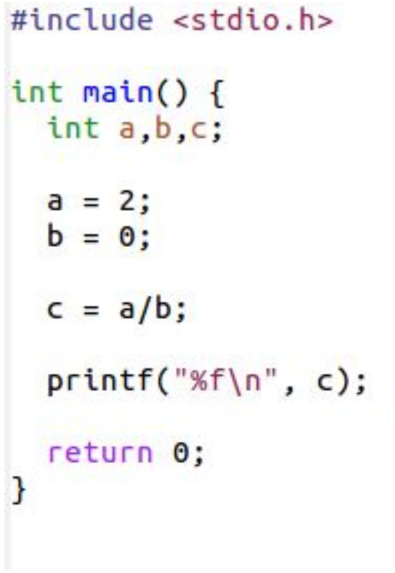


```c
int main() {
  float a,sum;
  int n,k;

  sum = 0;
  k = 0;
  while (1) {
    scanf("%f", &a);

    if (a < 0)
      break;

    sum = sum + a;
    k++;
  }


  printf("average=%f\n", sum/k);

  return 0;
}
```



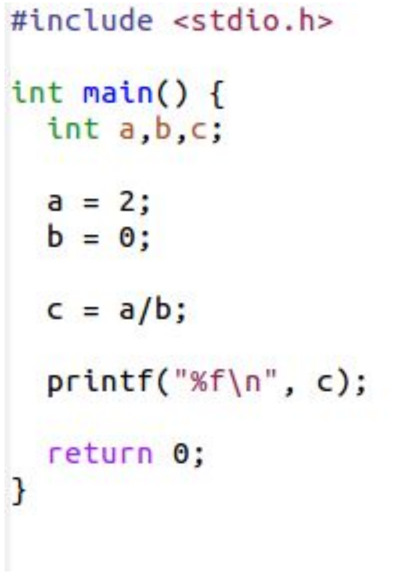
```c
int main() {
  float a,sum;
  int n,k;

  sum = 0;
  k = 0;

  scanf("%f", &a);
  while (a >= 0) {
    sum = sum + a;
    k++;

    scanf("%f", &a);
  }

  if (k == 0)
    puts("No grades entered");
  else
    printf("average=%f\n", sum/k);

  return 0;
}
```

Divide by zero!

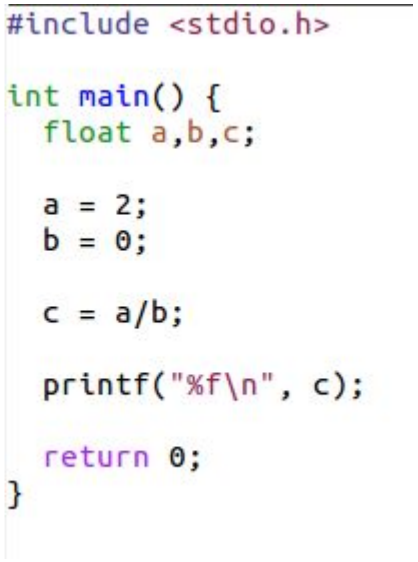


```c
#include <stdio.h>

int main() {
  int a,b,c;

  a = 2;
  b = 0;

  c = a/b;

  printf("%f\n", c);

  return 0;
}
```

Divide by zero!

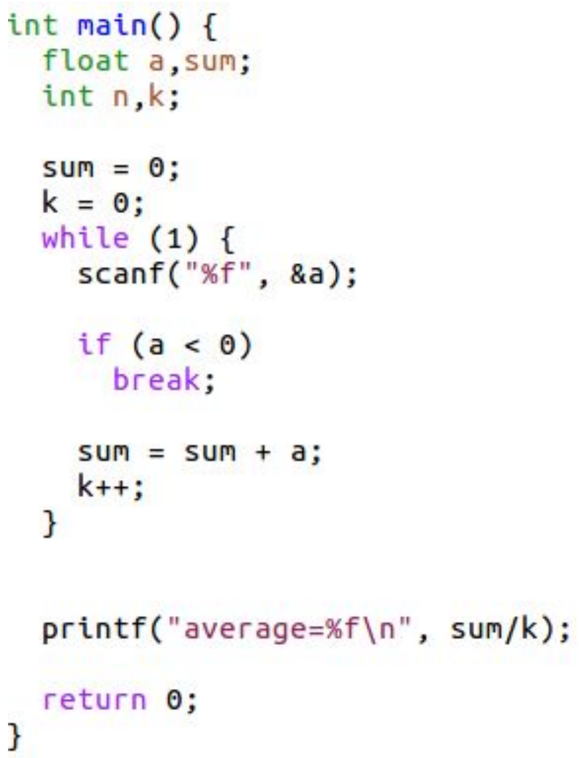


```c
#include <stdio.h>

int main() {
  int a,b,c;

  a = 2;
  b = 0;

  c = a/b;

  printf("%f\n", c);

  return 0;
}
```



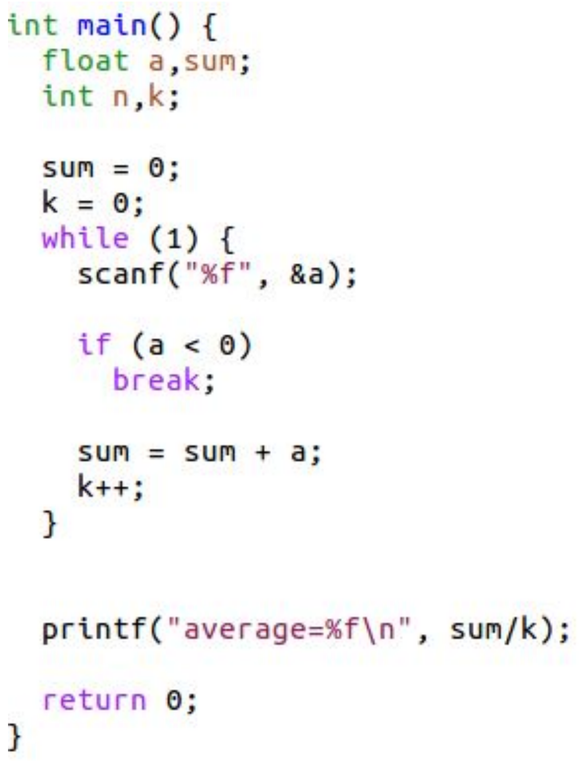
```c
#include <stdio.h>

int main() {
  float a,b,c;

  a = 2;
  b = 0;

  c = a/b;

  printf("%f\n", c);

  return 0;
}
```

printf floating point format

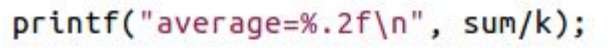


```c
int main() {
  float a,sum;
  int n,k;

  sum = 0;
  k = 0;
  while (1) {
    scanf("%f", &a);

    if (a < 0)
      break;

    sum = sum + a;
    k++;
  }


  printf("average=%f\n", sum/k);

  return 0;
}
```

printf floating point format

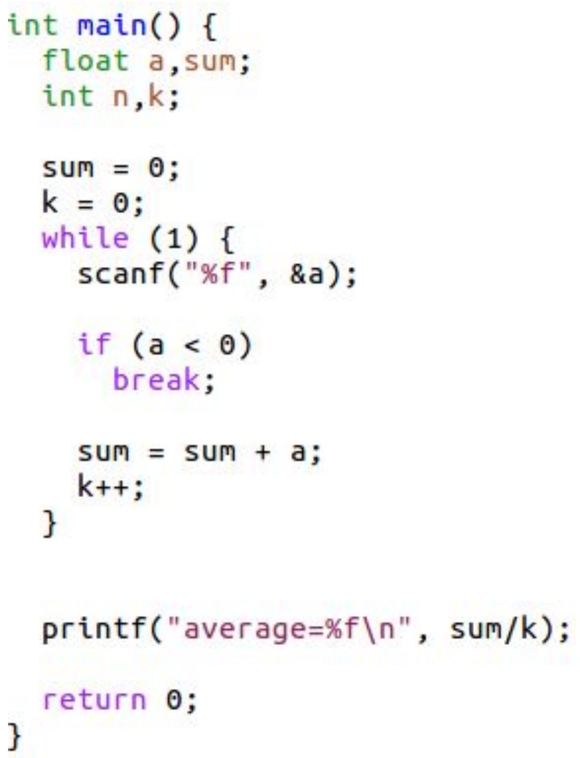


```
int main() {
  float a,sum;
  int n,k;

  sum = 0;
  k = 0;
  while (1) {
    scanf("%f", &a);

    if (a < 0)
      break;

    sum = sum + a;
    k++;
  }


  printf("average=%f\n", sum/k);

  return 0;
}
```

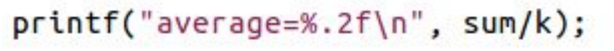


```
printf("average=%.2f\n", sum/k);
```

printf floating point format



```c
int main() {
  float a,sum;
  int n,k;

  sum = 0;
  k = 0;
  while (1) {
    scanf("%f", &a);

    if (a < 0)
      break;

    sum = sum + a;
    k++;
  }


  printf("average=%f\n", sum/k);

  return 0;
}
```



```c
printf("average=%.2f\n", sum/k);
```

printf floating point format

```
int main() {
  float a,sum;
  int n,k;

  sum = 0;
  k = 0;
  while (1) {
    scanf("%f", &a);

    if (a < 0)
      break;

    sum = sum + a;
    k++;
  }


  printf("average=%f\n", sum/k);

  return 0;
}
```

```
printf("average=%8.2f\n", sum/k);
```

printf floating point format

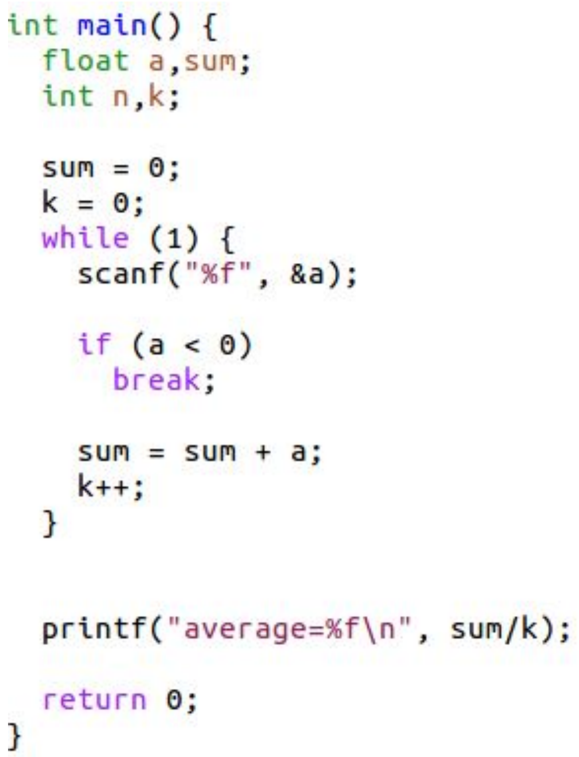
```
int main() {
  float a,sum;
  int n,k;

  sum = 0;
  k = 0;
  while (1) {
    scanf("%f", &a);

    if (a < 0)
      break;

    sum = sum + a;
    k++;
  }


  printf("average=%f\n", sum/k);

  return 0;
}
```

https://www.cprogramming.com/tutorial/printf-format-strings.html

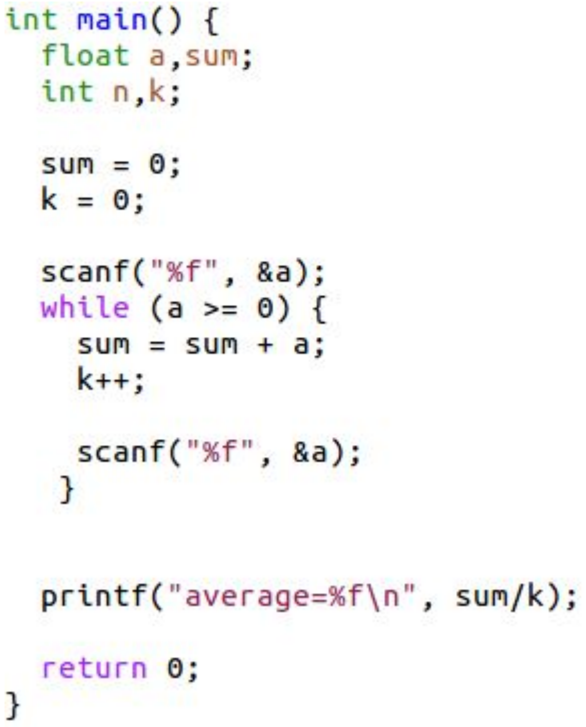
https://en.wikipedia.org/wiki/Printf_format_string

```
printf("average=%8.2f\n", sum/k);
```

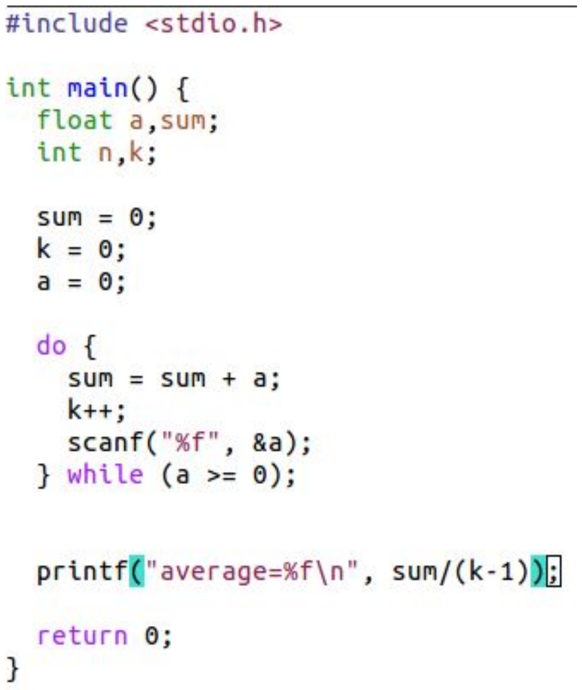
Remember from last session (Average score)



```c
int main() {
  float a,sum;
  int n,k;

  sum = 0;
  k = 0;
  while (1) {
    scanf("%f", &a);

    if (a < 0)
      break;

    sum = sum + a;
    k++;
  }


  printf("average=%f\n", sum/k);

  return 0;
}
```



```c
int main() {
  float a,sum;
  int n,k;

  sum = 0;
  k = 0;

  scanf("%f", &a);
  while (a >= 0) {
    sum = sum + a;
    k++;

    scanf("%f", &a);
   }


  printf("average=%f\n", sum/k);

  return 0;
}
```



```c
#include <stdio.h>

int main() {
  float a,sum;
  int n,k;

  sum = 0;
  k = 0;
  a = 0;

  do {
    sum = sum + a;
    k++;
    scanf("%f", &a);
  } while (a >= 0);


  printf("average=%f\n", sum/(k-1));

  return 0;
}
```

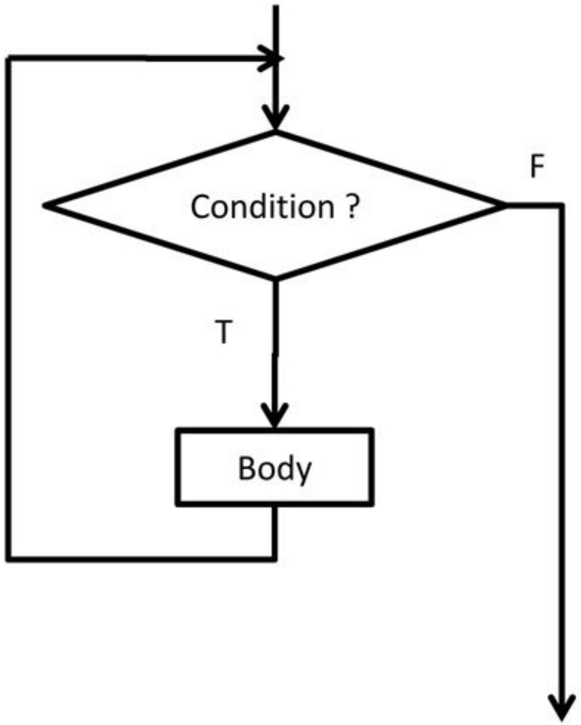
do-while loop

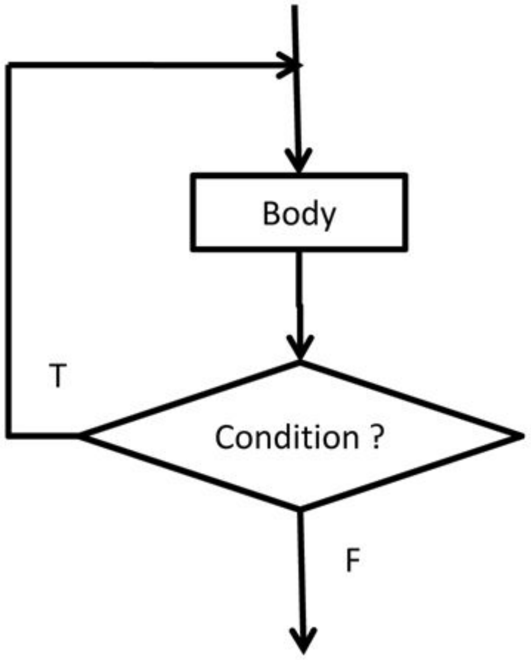
```
do {


} while (CONDITION);
```

While versus Do-While Loops

```
while( condition )
        body;
```

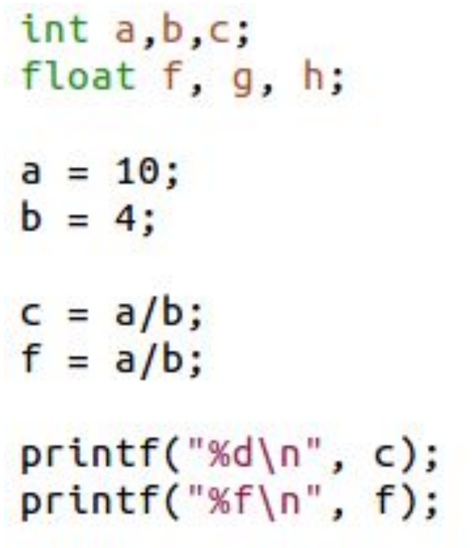
```
do {
        body;
} while( condition );
```





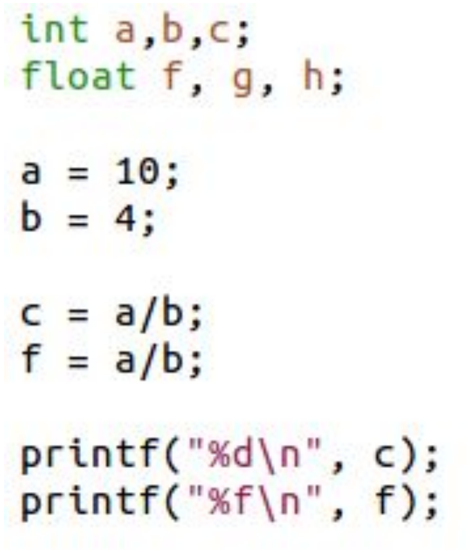
https://www.cs.uic.edu/~jbell/CourseNotes/C_Programming/Looping.html

Integer division

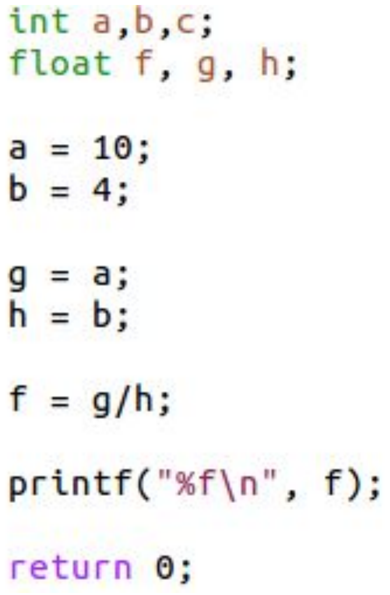


```c
int a,b,c;
float f, g, h;

a = 10;
b = 4;

c = a/b;
f = a/b;

printf("%d\n", c);
printf("%f\n", f);
```

Integer division

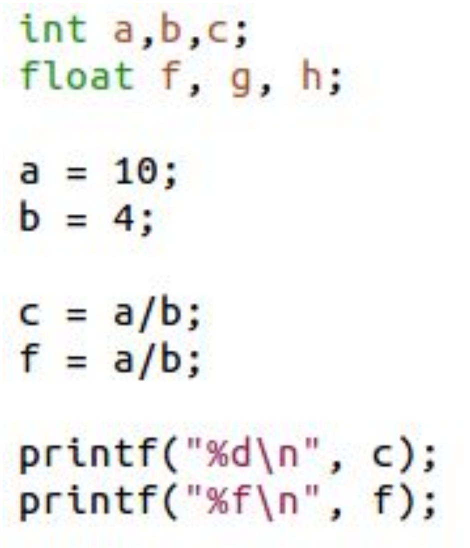


```c
int a,b,c;
float f, g, h;

a = 10;
b = 4;

c = a/b;
f = a/b;

printf("%d\n", c);
printf("%f\n", f);
```

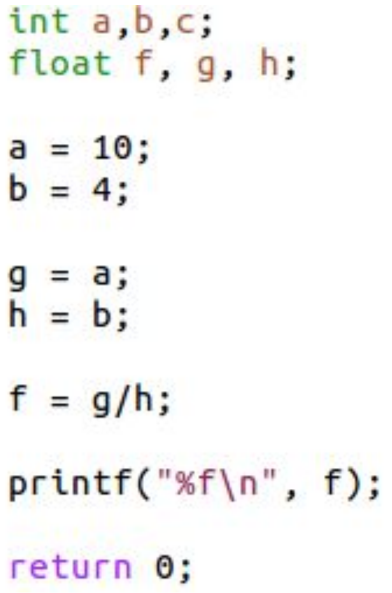


```c
int a,b,c;
float f, g, h;

a = 10;
b = 4;

g = a;
h = b;

f = g/h;

printf("%f\n", f);

return 0;
```

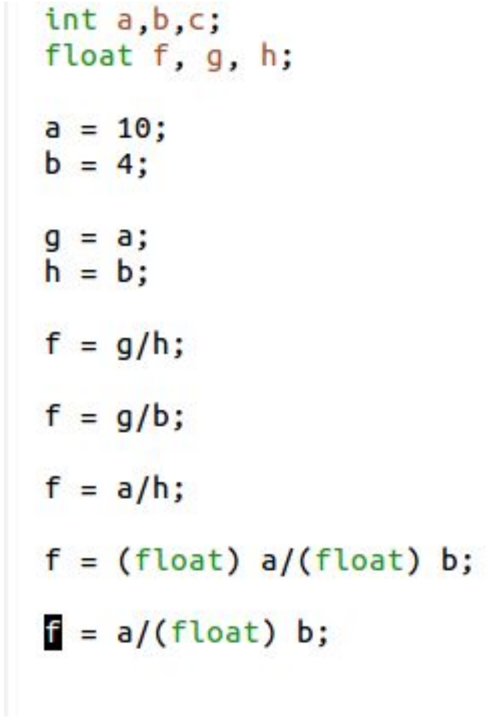
Integer division



```
int a,b,c;
float f, g, h;

a = 10;
b = 4;

c = a/b;
f = a/b;

printf("%d\n", c);
printf("%f\n", f);
```

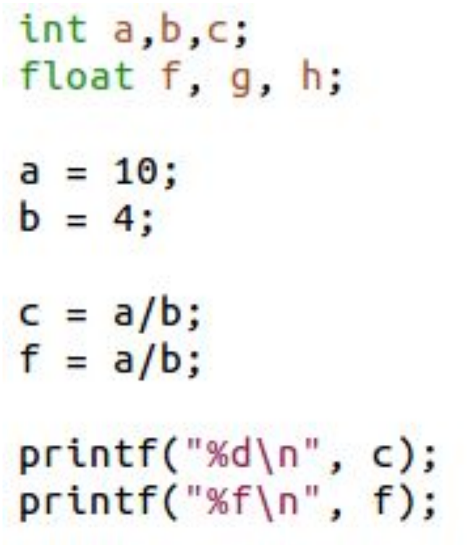


```
int a,b,c;
float f, g, h;

a = 10;
b = 4;

g = a;
h = b;

f = g/h;

printf("%f\n", f);

return 0;
```

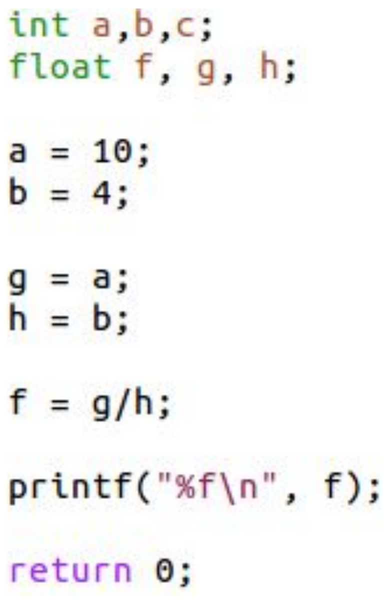


```
int a,b,c;
float f, g, h;

a = 10;
b = 4;

g = a;
h = b;

f = g/h;

f = g/b;

f = a/h;

f = (float) a/(float) b;

f = a/(float) b;
```

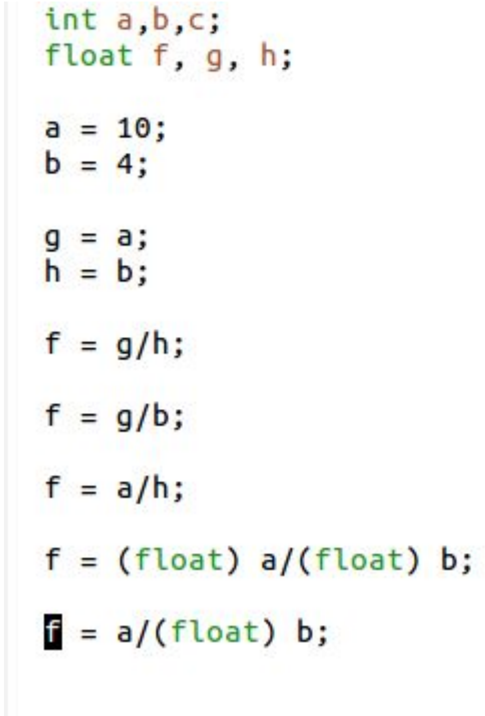
Integer division



```c
int a,b,c;
float f, g, h;

a = 10;
b = 4;

c = a/b;
f = a/b;

printf("%d\n", c);
printf("%f\n", f);
```



```c
int a,b,c;
float f, g, h;

a = 10;
b = 4;

g = a;
h = b;

f = g/h;

printf("%f\n", f);

return 0;
```



```c
int a,b,c;
float f, g, h;

a = 10;
b = 4;

g = a;
h = b;

f = g/h;

f = g/b;

f = a/h;

f = (float) a/(float) b;

f = a/(float) b;
```

Example: power

Write a program reading a float number "a" and a positive integer "b" and printing a^b

Example: power

```
float a;
int b;
float p;

scanf("%f %d", &a,&b);

p = 1;
while (b > 0) {
  p *= a;
  b--;
}

printf("%f\n",p);
```

Example: factorial

Write a program readig an integer "n" and printing its factorial (n!).

operators - precedence

```
a + b * c
```

operators - precedence

```
a + b * c

a + (b * c)
```

operators - precedence

```
a / b - d * a
```

operators - precedence

```
a / b - d * a
(a / b) - (d * a)
```

operators - precedence

```
a * -b - -c * d
```

operators - precedence

```
a * -b - -c * d
a * (-b) - (-c) * d
```

operators - precedence

```
a * -b - -c * d
(a * (-b)) - ((-c) * d)
```

operators - precedence

```
a / b / c
```

operators - precedence

```
(a / b) / c
```

operators - precedence

```
a - b + c

a + b - c

a * b / c

a / b * c
```

operators - precedence

```
a - b + c   ⟹   (a - b) + c
a + b - c   ⟹   (a + b) - c
a * b / c   ⟹   (a * b) / c
a / b * c   ⟹   (a / b) * c
```

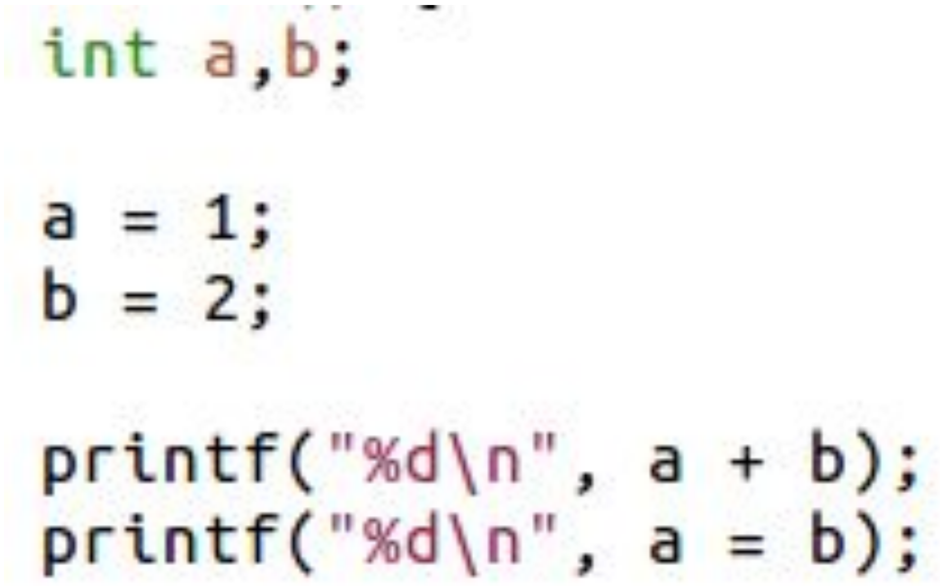
Assignment operators

op	usage	equivalent
`+=`	`a += b`	`a = a + b`
`-=`	`a -= b`	`a = a - b`
`*=`	`a *= b`	`a = a * b`
`/=`	`a /= b`	`a = a / b`

increment and decrement

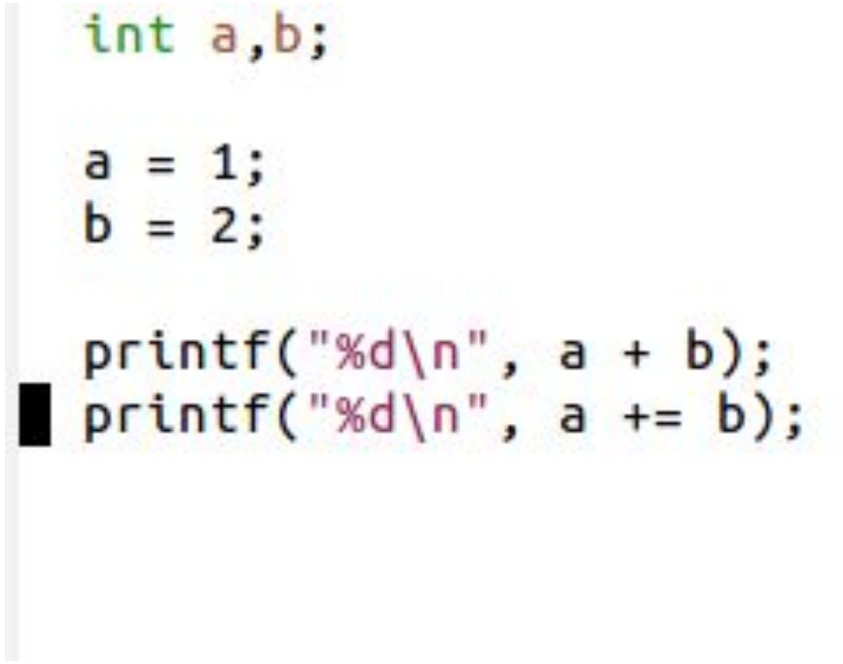
op	usage	equivalent
`++`	`a++`	`a = a + 1 (*)`
`++`	`++a`	`a = a + 1`
`--`	`a--`	`a = a - 1 (*)`
`--`	`--a`	`a = a - 1`

Assignment as an operator



```
int a,b;

a = 1;
b = 2;

printf("%d\n", a + b);
printf("%d\n", a = b);
```

Assignment as an operator



```
int a,b;

a = 1;
b = 2;

printf("%d\n", a + b);
printf("%d\n", a += b);
```

operators - assignment

```
a = b = c
```

operators - assignment

```
a = (b = c)
```

operators associativity

```
a = b = c      ⟹    a = (b = c)
a - b - c      ⟹    (a - b) - c
```

operators associativity

right to left

```
a = b = c      ⟹     a = (b = c)

a - b - c      ⟹     (a - b) - c
```

left to right

operators - assignment

```
a = b = c = d = e;
```

operators - assignment

```
- -  -  a
```

operators - assignment

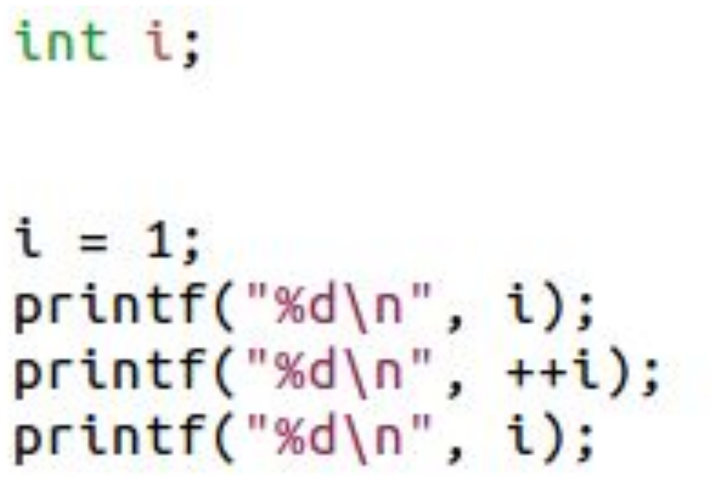
```
- (- (-a))
```

operators - assignment

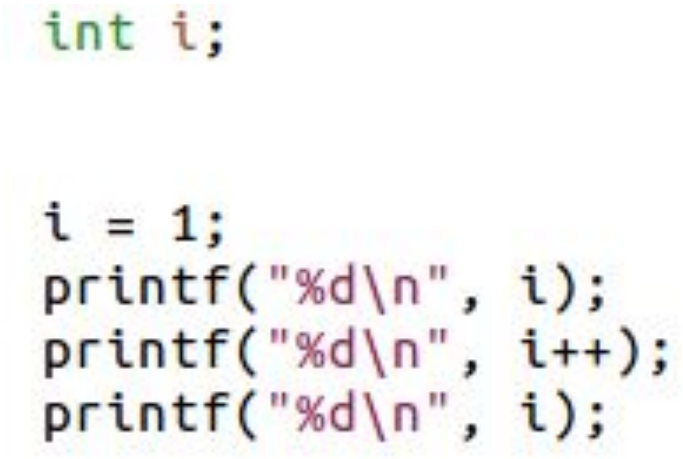
$\boxed{\text{right to left}}$

```
- (- (-a))
```

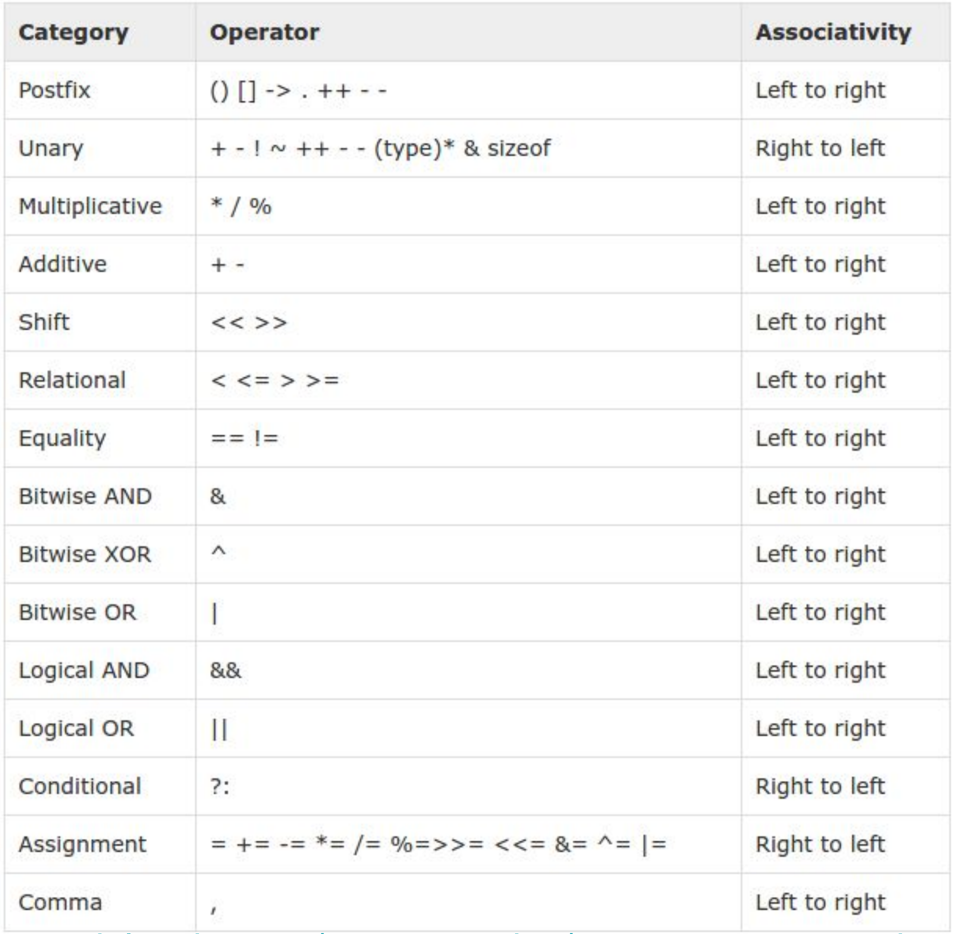
increment and decrement



```
int i;


i = 1;
printf("%d\n", i);
printf("%d\n", ++i);
printf("%d\n", i);
```



```
int i;


i = 1;
printf("%d\n", i);
printf("%d\n", i++);
printf("%d\n", i);
```



Category	Operator	Associativity
Postfix	() [] -> . ++ - -	Left to right
Unary	+ - ! ~ ++ - - (type)* & sizeof	Right to left
Multiplicative	* / %	Left to right
Additive	+ -	Left to right
Shift	<< >>	Left to right
Relational	< <= > >=	Left to right
Equality	== !=	Left to right
Bitwise AND	&	Left to right
Bitwise XOR	^	Left to right
Bitwise OR	|	Left to right
Logical AND	&&	Left to right
Logical OR	||	Left to right
Conditional	?:	Right to left
Assignment	= += -= *= /= %=>>= <<= &= ^= |=	Right to left
Comma	,	Left to right

https://www.tutorialspoint.com/cprogramming/c_operators_precedence.htm

operators - comparison

```
a + b >= c * d
```

operators - comparison

```
a > b + c && k == d
```

operators - comparison

```
10 > 16 > 20
```

operators - comparison

```
(10 > 16) > 20
```

operators - comparison

```
1 > 20
```