

HashMaps

```
use std::collections::HashMap;

let mut h: HashMap<i64,i64> = HashMap::new();

h.contains_key(&(n,k))    check whether key is present
h.insert((n,k), res);    insert new value
*h.get(&(n,k)).unwrap()  retrieve value
```

Variables and constants

```
const N: i32 = 5          (Global) constant
let v = vec![3, 1, 4]    Define and initialize vector
let zero_vec = vec![0; len]
let mut p : [usize; 9] = [0; 9];
```

Primitive types

bool	Boolean
char	character
f32, f64	32-bits, 64-bits floats
i64, i32, i16, i8	signed 16- ... integers
u64, u32, u16, u8	unsigned 16-bits, ... integers
isize	pointer-sized signed integers
usize	pointer-sized unsigned integers

Processor directives

```
#![feature(iterator_step_by)]
| for n in (0..100).step_by(2) { ... } iterates over even numbers

#![feature(inclusive_range_syntax)]
| 1..=n is a range which includes both 1 and n
```

Boolean operators

Operator	Syntax
And	a & b or a && b
Or	a . b or a .. b
And	a ^ b
Not	!a

Conversions

```
To convert say an n: i32 to a u64 use n as u64

To convert a number x to a string use x.to_s tring()

To convert s: String* to x.pars e().un wrap()

x.unwr ap_or( def aul t_v alue)
```

Formatting

```
println!("{}", x);
println!("{:010b}", x);    binary, 10 digits, print leading zeros
format!("Hello");         "Hello"
format!("Hello, {}!", "world"); "Hello, world!"
format!("The number is {}", 1); "The number is 1"
format!("{:?}", (3, 4));   "4"
format!("{value}", value=4); "1 2"
format!("{:04}", 42);     "0042" with leading zeros
```

Vectors and iterators

```
(0..16usize).map(|x| x.count_ones() as usize).collect()
```

Assertions

```
let x = 42;
assert!(x == 42);
assert_eq!(x, 42);
```

Use the macro `assert!` to check for a boolean true, and `assert_eq!` for the equality of two expressions.



By **Reiner Martin** (Reiner Martin)
cheatography.com/reiner-martin/

Not published yet.
 Last updated 25th January, 2018.
 Page 1 of 1.

Sponsored by **CrosswordCheats.com**
 Learn to solve cryptic crosswords!
<http://crosswordcheats.com>