# 数据类型

本章讨论讨论的数据类型均为各语言的基本数据类型的定义形式和类型相关，即语言内建类型或储存于语言栈上的数据类型（不包括OOP相关）

## C++

## Java

## Python

## Go

## Rust

### 形式定义

在rust 中存在变量和常量，其变量的定义使用绑定的概念，形式如下:

let [mut] variable\_name [:variable\_type] [= value ;]

说明如下

mut 代表该变量不能改变(类型和值)

常量的定义形式如下

const constant\_name: constant\_type = value;

其中常量的类型式必须明确定义

在rust中固定变量和常量有明显的区别

1. 常量必须有明确类型定义，而变量可以通过绑定数值，进而有类型推断得到，所以不是必须明确定义
2. 常量的定义定义可以式一个常量表达式，而不能式一个函数调用或者其他只能在运行时计算结果的表达式
3. 常量的生命周期为整个程序

### 数据类型

rust中的基本数据类型如下

1. scalar(基本类型)
   1. integer
   2. floatpoint
   3. boolean
   4. characters
2. compound(复合类型)
   1. tuple
   2. array
3. string  
   1. str
   2. slice
4. other
   1. pointer
   2. reference

具体的内容如下

1. integer(整形): rust中的整形按照有无符号分为2大类

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| signed | **i8** | **i16** | **i32** | **i64** | **i128** | **isize** |
| unsigned | **u8** | **u16** | **u32** | **u64** | **u128** | **usize** |

其中\*size取决于系统，在32位系统上其内部实现位\*32，在64位系统上其内部实现位\*64。

其初始化形式为

| Number LIterals | Example |
| --- | --- |
| Decimal | 98\_222 |
| Hex | 0xff |
| Octal | 0o77 |
| Binary | 0b1111\_0000 |
| Byte（u8 only） | b'A' |

1. float(浮点): rust 中浮点按照长度划分为2种

|  |  |
| --- | --- |
| 单精度 | f32 |
| 双精度 | f64 |

1. bool(布尔): true 和false，rust 中boolean 类型只有1byte,bool类型可以强制转换为整数(false as i32)，其值为true 对应1 ，false对应0
2. char(字符类型): rust的字符使用4byte 的Unicode 编码，绑定时使用单引号。
3. tuple: tuple 的定义形式为

let tuple\_name [:(tuple\_element\_type …)] [= value;]

tuple 类型长度固定，类型一旦声明就不能随便改变

内部元素的访问形式可以使用

* 1. 解构：

*let tup = (1,’2’,3.0);*

*let (x,y,\_) = tup;*

* 1. 序号 : 越界错误出现在编译时

tup.0 ,tup.1

1. array: rust的array类型并不对应于其他语言的同名类型(比如Python),vector才是对应的类型，

array类型需要注意的事项如下:

* 1. array 类型固定长度，一旦定义就不能改变
  2. array的大小不能超过32
  3. 如果array的元素是Copy ，array就是Copy；如果array的元素式Clone，array就是Clone

1. 就

## Scala

## Clojure

## Kotlin