1、NSRange类

```
/*
 typedef struct _NSRange {
    NSUInteger location;
    NSUInteger length;
 } NSRange;
 */

// NSRange类表示为一个范围，有两个参数location和length
// location表示范围的起始位置
// length表示范围的长度

NSRange r1;

//给NSRange赋值的不同方式
// 1)
r1.length = 3;
r1.location = 5;

// 2)
r1 = (NSRange){1,2};
NSRange r2 = {4,5};

// 3)
NSRange r3 = {.location = 9, .length = 10};

// 4)
NSRange r4 = NSMakeRange(5, 6);

NSLog(@"location = %ld, length = %ld",r4.location, r4.length);
NSLog(@"%@",NSStringFromRange(r4));
```

2、NSString创建，及文件读写

```
// 1) 字符串的创建方式，及存储位置
NSString *str1 = @"mark";//常量区
NSString *str2 = [NSString stringWithFormat:@"%@", @"jack"];//堆区
NSString *str3 = [[NSString alloc] initWithFormat:@"My age is %d", 10];//常量区

NSLog(@"str1 = %@, at %p", str1, str1);//str1 = mark, at 0x100001048
NSLog(@"str2 = %@, at %p", str2, str2);//str2 = jack, at 0x6b63616a45
NSLog(@"str3 = %@, at %p", str3, str3);//str3 = My age is 10, at 0x1004000a0
```

```
// 2) 将字符串写入从文件中
NSError *err;
NSString *str = @"123";
BOOL isright;

//写入文件方法writeToFile
isright = [str writeToFile:@"/Users/caokun/Desktop/str.txt"
                atomically:YES encoding:NSUTF8StringEncoding error:&err];

//判断是否写入成功，方式1
if (err != nil) {
    NSLog(@"写入失败, %@",err);
}else{
    NSLog(@"写入成功");
}
//方式2
if(isright){
    NSLog(@"写入成功");
}else{
    NSLog(@"写入失败, %@",err);
}
```

```
// 3)从文件中读取内容

//从文件中读取方法stringWithContentsOfFile
NSString *str4 = [NSString stringWithContentsOfFile:@"/Users/caokun/Desktop/str.txt"
    encoding:NSUTF8StringEncoding error:&err];

if (err != nil) {
    NSLog(@"读入失败, %@",err);
}else{
    NSLog(@"读入成功");
    NSLog(@"str4 = %@", str4);
}
```

3、NSURL,OC中处理URL的类，用URL读写

```
NSString *str1 = @"$1000000";

//定义一个URL类型，URL (统一资源定位符)
/*
    file://     本地文件
    ftp://      ftp文件
    http://     http文件
    https://    加密文件
 */

// 1) 通过URLWithString 构建一个URL
//可以构建本地路径的URL   [NSURL URLWithString:@"file:///users/caokun/desktop/1.txt"];
//可以构建调用手机系统的程序   [NSURL URLWithString:@"tel://10086"];
NSURL *url = [NSURL URLWithString:@"file:///users/caokun/desktop/1.txt"];

// 2) 通过FileURLwithPath 构建一个URL
NSURL *url1 = [NSURL fileURLWithPath:@"/users/caokun/desktop/1.txt"];

//用URL方式写入文件
NSString *str5;
if([str5 writeToURL:url1 atomically:YES encoding:NSUTF8StringEncoding error:nil]){
    NSLog(@"写入成功");
}
//用URL方式读取文件
NSError *err1;
str5 = [NSString stringWithContentsOfURL:url1 encoding:YES error: &err1];
if(err1 == nil){
    NSLog(@"%@",str5);
}
```

4、NSString类，判断字符串是否相等: isEqualToString: （NSString *）

```
NSString *str1 = @"123";//常量区
NSString *str2 = [NSString stringWithFormat:@"123"];//堆区

if(str1 == str2){
    NSLog(@"字符串相等");
}else{
    NSLog(@"字符串不相等");
}
//输出不相等,因为地址位置不同,

//so, 判断字符串相等用isEqualToString方法
if ([str1 isEqualToString:str2]) {
    NSLog(@"equal");
}
```

5、NSString类，判断前缀、后缀、字符串包含

```
NSString *str1 = @"http://www.google.com";
//检测前缀
if([str1 hasPrefix:@"http://"]){
    NSLog(@"前缀是http://");
}

NSString *str2 = @"xxxx.jpg";
//检测后缀
if([str2 hasSuffix:@".jpg"]){
    NSLog(@"后缀是.jpg");
}
```

```
NSString *str3 = @"http://www.google.com";
NSString *str4 = @"goog";
//查找字符串str4在str3中出现的位置
/*
 查找到是，返回一个NSRange类型的数，包括首地址location和str4的长度
 查找不到时，location返回最大long整形数，length返回0
 */

NSRange range;
range = [str3 rangeOfString:str4];
if(range.length){
    NSLog(@"%lu, %lu",range.location,range.length);
}else{
    NSLog(@"没有查找到");
}
```

6、NSString类，字符串的裁剪和替换

```
NSString *str1 = @"http://www.google.com";

//从某位置开始，到字符串结尾
NSString *str2 = [str1 substringFromIndex:5];
//从字符串头开始，到某位置
NSString *str3 = [str1 substringToIndex:5];
//根据某范围截取
NSString *str4 = [str1 substringWithRange:(NSRange){3,7}];
//将字符串中某一子字符串，替换成其他字符串
NSString *str5 = [str1 stringByReplacingOccurrencesOfString:@"oo" withString:@"2"];


NSLog(@"%@, %@, %@, %@",str2, str3, str4, str5);
```

7、NSString类，跟其他类型转换

```
NSString *str1 = @"23";
NSString *str2 = @"5.67";

//把NSString对象转换成数值
int a = [str1 intValue];
float b = [str2 floatValue];
double c = [str2 doubleValue];

NSLog(@"%d, %.2f, %.2f",a ,b ,c);

//把NSSstring对象转换成char*类型
char *str3 = [str2 UTF8String];
printf("%s\n",str3);

//把char*类型转换成NSString类型
char *s = "abcdefg";
NSString *str4 = [NSString stringWithUTF8String:s];

printf("%s\n",s);
NSLog(@"%@",str4);
```

8、可变字符串NSMutableString常用方法

```
NSMutableString *str = [NSMutableString stringWithFormat:@"boy"];

//添加一个字符串
[str appendString:@"+girl"];

//按某种格式添加一个字符串
int a = 100;
[str appendFormat:@",a = %d",a];

//删除字符串一部分内容
[str deleteCharactersInRange:NSMakeRange(1, 5)];

//插入字符串
[str insertString:@"1234" atIndex:2];

//替换字符串
[str replaceCharactersInRange:NSMakeRange(1, 2) withString:@"00000"];

//将字符串制空
str.string = @"";

NSLog(@"%@",str);
```

9、数组类NSArray

```
//NSArray注意事项
//必须存储OC对象，不能存储非OC对象，比如int\float\double\char\enum\struct

// 1) 创建一个空数组
NSArray *array1 = [NSArray array];
// 2) 创建数组，只有一个元素
NSArray *array2 = [NSArray arrayWithObject:@"1"];
// 3) 创建数组，有多个元素
NSArray *array3 = [NSArray arrayWithObjects:@"one",@"two",[NSNull null],nil];
// 4) 调用对象方法，创建数组
NSArray *array4 = [[NSArray alloc] initWithObjects:@"one",@"two",@1,nil];
// 5) 用一个数组创建另一个数组
NSArray *array5 = [NSArray arrayWithArray:array3];
// 按%@格式，可以打印数组所有值
NSLog(@"%@",array3);
```

```
//简化方式定义数组
NSArray *arr = @[@"1", @"2", @3, @"4"];
NSLog(@"%@",arr);

id str = arr[1];
NSLog(@"%@",str);
```

```
//数组遍历
//方式1
for(int i = 0; i < arr.count; i++){
    NSLog(@"%@",arr[i]);
}

//方式2
for(NSString *str in arr){
    NSLog(@"%@",str);
}

//方式3
[arr enumerateObjectsUsingBlock:^(id  _Nonnull obj, NSUInteger idx, BOOL * _Nonnull stop){
        if(idx == 2){//判断停止
            *stop = YES;
        }else{
            NSLog(@"idx == %ld, obj == %@",idx, obj);
        }
}];
```

```
//数组写入到文件
if([arr writeToFile:@"/users/caokun/desktop/arr.plist" atomically:YES]){
    NSLog(@"写入成功");
}

//读出数组文件
NSArray *arrWrite = [NSArray arrayWithContentsOfFile:@"/users/caokun/desktop/arr.plist"];
NSLog(@"%@",arrWrite);
```

```
//将数组合成字符串
NSArray *arr1 = @[@1,@2,@3,@4];
NSString *str1 = [arr1 componentsJoinedByString:@"-"];
NSLog(@"%@",str1);

//将字符串分割成数组
NSString *str2 = @"400-800-8820";
NSArray *arr2 = [str2 componentsSeparatedByString:@"-"];
NSLog(@"%@",arr2);
```

10、可变数组NSMutableString介绍

```
// 1) 创建可变数组
//空数组
NSMutableArray *arr3 = [NSMutableArray array];
//单个元素
NSMutableArray *arr2 = [NSMutableArray arrayWithObject:@"one"];
//多个元素
NSMutableArray *arr1 = [NSMutableArray arrayWithObjects:@"one",@"two",@"there",nil];
//指定数组大小
NSMutableArray *arr4 = [NSMutableArray arrayWithCapacity:5];

// 2) 向数组添加元素
[arr1 addObject:@"number"];
//插入元素到指定位置
[arr1 insertObject:@"ten" atIndex:0];

// 3)删除元素
//根据内容删除
[arr1 removeObject:@"number"];
//根据位置删除
[arr1 removeObjectAtIndex:0];

// 4)修改元素
[arr1 replaceObjectAtIndex:0 withObject:@"two"];
arr1[0] = @"five";
NSLog(@"%@",arr1);

// 5)查找元素
BOOL isSearch = [arr1 containsObject:@"there"];
NSLog(@"%d",isSearch);
NSLog(@"%@",arr1);

// 6)交换元素
[arr1 exchangeObjectAtIndex:0 withObjectAtIndex:1];
NSLog(@"%@",arr1);
```

11、字典类NSDictionary介绍和使用

介绍：通过一个Key，对应一个Value

```
// 1) 创建字典
//空字典
NSDictionary *dic1 = [NSDictionary dictionary];
//单个元素
NSDictionary *dic2 = [NSDictionary dictionaryWithObject:@"zhaosi" forKey:@"1"];
//多个元素
NSDictionary *dic3 = [NSDictionary dictionaryWithObjectsAndKeys:@"zhaosi",@"2",@"zhangsan",@"2", nil];
//快速创建字典
NSDictionary *dic4 = @{@"1":@"zhaosi", @"2":@"zhangsan", @"3":@"wangwu"};

// 2)获取个数
int num = dic4.count;

// 3) 通过Key查找Value
NSString *str1 =  [dic4 objectForKey:@"2"];
NSLog(@"%@",str1);

// 4)字典遍历
//方式1
for(NSString *key in dic4){
    NSLog(@"key = %@, value = %@",key ,[dic4 objectForKey:key]);
}

//方式2
[dic4 enumerateKeysAndObjectsUsingBlock:^(id  _Nonnull key, id  _Nonnull obj, BOOL * _Nonnull stop) {
    NSLog(@"key = %@, value = %@",key ,obj);
}];
```

```
// 5) 简写获取方式
str1 = dic4[@"1"];
NSLog(@"%@",str1);

// 6) 把字典保存到文件
if([dic4 writeToFile:@"/users/caokun/desktop/dic1.plist" atomically:YES]){
    NSLog(@"写入成功");
}

// 7) 从文件中取出字典
NSDictionary *dic5 = [NSDictionary dictionaryWithContentsOfFile:@"/users/caokun/desktop/dic1.plist"];
NSLog(@"%@",dic5);

// 8) 多层类型数据字典的调用
NSArray *lnArr = [NSArray arrayWithObjects:@"dalian", @"liaoning",nil];
NSArray *hbArr = [NSArray arrayWithObjects:@"shijiazhuang", @"cangzhou",nil];
NSDictionary *dCitys = [NSDictionary dictionaryWithObjectsAndKeys:lnArr,@"ln",hbArr,@"hb", nil];
//写入文件
[dCitys writeToFile:@"/users/caokun/desktop/dic2.plist" atomically:YES];
//读取文件，并遍历
NSDictionary *readCitys = [NSDictionary dictionaryWithContentsOfFile:@"/users/caokun/desktop/dic2.plist"];
[readCitys enumerateKeysAndObjectsUsingBlock:^(id  _Nonnull key, id  _Nonnull obj, BOOL * _Nonnull stop) {
    for(NSString *strr in obj){
        NSLog(@"%@",strr);
    }
}];
//读取单个数据
NSArray *arr5 = readCitys[@"hb"];
NSLog(@"%@",arr5[0]);
```

12、文件类NSFileManager的介绍和使用


```
// 1) 创建字典
//空字典
NSDictionary *dic1 = [NSDictionary dictionary];
//单个元素
NSDictionary *dic2 = [NSDictionary dictionaryWithObject:@"zhaosi" forKey:@"1"];
//多个元素
NSDictionary *dic3 = [NSDictionary dictionaryWithObjectsAndKeys:@"zhaosi",@"2",@"zhangsan",@"2", nil];
//快速创建字典
NSDictionary *dic4 = @{@"1":@"zhaosi", @"2":@"zhangsan", @"3":@"wangwu"};

// 2)获取个数
int num = dic4.count;

// 3) 通过Key查找Value
NSString *str1 =  [dic4 objectForKey:@"2"];
NSLog(@"%@",str1);

// 4)字典遍历
//方式1
for(NSString *key in dic4){
    NSLog(@"key = %@, value = %@",key ,[dic4 objectForKey:key]);
}

//方式2
[dic4 enumerateKeysAndObjectsUsingBlock:^(id  _Nonnull key, id  _Nonnull obj, BOOL * _Nonnull stop) {
    NSLog(@"key = %@, value = %@",key ,obj);
}];
```


```
//获取文件信息
NSFileManager *fm = [NSFileManager defaultManager];
NSString *filePath = @"/users/caokun/desktop/dic1.plist";

// 1) 获取文件的属性
NSDictionary *dic = [fm attributesOfItemAtPath:filePath error:nil];
NSLog(@"%@",dic);
NSLog(@"%@, %@",dic[@"NSFileOwnerAccountName"]);

// 2)获取指定目录下的文件(不获取子目录路径)
NSString *filePath2 = @"/users/caokun/desktop";
NSArray *subPath1 = [fm contentsOfDirectoryAtPath:filePath2 error:nil];
NSLog(@"%@",subPath1);

// 3) 获取指定目录下的所有文件及子目录
//使用递归模式，获取当前目录及子目录所有文件和文件夹
NSArray *subPath2 = [fm subpathsAtPath:filePath2];
NSLog(@"%@",subPath2);
```

```
//文件操作
NSString *str = @"天空刚下过一场雨，路上的行人不多。";
NSString *filePath1 = @"/users/caokun/desktop/boy.txt";
NSData *data = [str dataUsingEncoding:NSUTF8StringEncoding];
NSFileManager *fm = [NSFileManager defaultManager];

// 1) 创建文件aaa
BOOL isYes = [fm createFileAtPath:filePath1 contents:data attributes:nil];
NSLog(@"%d",isYes);


NSString *fromPath = @"/users/caokun/desktop/aaa/bbb/girl.txt";
NSString *toPath = @"/users/caokun/desktop/ppp/pig.txt";
NSError *err;

// 2) 复制文件.注意：（此时的toPath路径，必须是以文件结尾".../pig.txt"，不能以文件夹结尾".../ppp")
isYes = [fm copyItemAtPath:fromPath toPath:toPath error:&err];

// 3) 移动文件
isYes = [fm moveItemAtPath:fromPath toPath:toPath error:&err];

// 4) 删除文件
isYes = [fm removeItemAtPath:toPath error:nil];
```

13、常见结构体

```
//常用结构体 NSPoint(点结构)、NSSize(大小结构)、NSRect(矩形结构)
//注意：以下这三组同义 NSPoint - CGPoint, NSSize - CGPoint, NSRect - CGRect

// 1) NSPoint(点结构)
NSPoint p;
p.x = 10;
p.y = 20;
//或者
p = NSMakePoint(10, 20);

// 2) NSSize(大小结构)
NSSize s;
s.height = 100;
s.width = 200;
//或者
s = NSMakeSize(200, 300);

// 3) NSRect(矩形结构)
NSRect r;
r.origin = p;
r.size = s;
//或者
r.origin.x = 20;
r.origin.y = 40;
r.size.height = 100;
r.size.width = 200;
//或者
r = NSMakeRect(20, 40, 100, 200);

// 4) 打印矩形结构，转换成字符串
NSLog(@"%@",NSStringFromRect(r));
```

14、数值类NSNumber的介绍和使用

```
//数值类NSNumber的介绍和使用

//定义整形NSNumber
int a = 10;
NSNumber *inta = [NSNumber numberWithInt:a];
//定义浮点型NSNumber
float f = 10.23;
NSNumber *floatObj = [NSNumber numberWithFloat:f];

//从NSNumber对象中取值
int sum = [inta intValue] + [floatObj floatValue];

//将NSNumber添加到数组中
NSMutableArray *arry = [NSMutableArray array];
[arry addObject:inta];
[arry addObject:floatObj];
//简洁的书写方法
[arry addObject:@(a)];
[arry addObject:@20];
[arry addObject:@YES];
```

15、数据类NSValue的介绍和使用

```
//NSValue的介绍和使用
//NSValue主要用来把指针、CGRect结构体等包装成OC对象，以便存储。
//NSValue类的目标是允许任意类型（int,float,char,pointer,struct,and Object ids等)
//           的数据结构能够被添加到集合里。

// 1) 包装基础数据
NSRect r1 = NSMakeRect(10, 20, 100, 200);
NSValue *val = [NSValue valueWithRect:r1];

NSMutableArray *arry = [NSMutableArray array];
[arry addObject: val];
[arry addObject: [NSValue valueWithRect:r1]];
[arry addObject: [NSValue valueWithRect:NSMakeRect(1, 2, 3, 4)]];

// 2) 包装自定义数据
typedef struct D{
    int year;
    int month;
    int day;
}MyDate;

MyDate d = {2016,3,21};

//以二进制方式存储成NSValue类型
NSValue *val1 = [NSValue valueWithBytes:&d objCType:@encode(MyDate)];

[arry removeAllObjects];
[arry addObject:val1];//添加到数组
NSValue *v = [arry firstObject];//从数组中取出

//从NSValue中取出结构体值getValue
MyDate res;
[val1 getValue:&res];
NSLog(@"%d, %d, %d",res.year,res.month,res.day);
```

16、时间类NSDate的介绍和使用

```
// 1) 时间类NSDate
NSDate *date = [NSDate date]; //当前时间
NSLog(@"%@",date);

// 2) 格式化显示时间
/*
 yyyy  年份
 MM    月份
 dd    天数
 HH    小时（24小时制）
 hh    小时（12小时制）
 mm    分钟
 ss    秒数
 */
NSDateFormatter *fm = [NSDateFormatter new];
//格式化时间书写格式
fm.dateFormat = @"yyyy/MM/dd HH:mm:ss";
NSString *str = [fm stringFromDate:date];
NSLog(@"%@",str);

//通过格式化创建NSDate
NSString *timeStr = @"1989-12-26 09:10:30";
NSDate *myDate = [fm dateFromString:timeStr];
NSLog(@"--%@",myDate);
```

```
// 3) 计算时间
NSTimeInterval t = 60*60*24;//间隔的秒数，可以为负值（代表以前的时间）
NSDate *tomorrowTime = [NSDate dateWithTimeIntervalSinceNow:t];
NSString *str2 = [fm stringFromDate:tomorrowTime];
NSLog(@"%@",str2);

// 4) 获取时间属性
NSCalendar *cal = [NSCalendar currentCalendar];
//属性是『按位或』组成的数值，所以要通过『|』来获取想要的属性
NSDateComponents *com = [cal components:NSCalendarUnitYear|NSCalendarUnitMonth|NSCalendarUnitDay|NSCalendarUnitHour|
    NSCalendarUnitMinute|NSCalendarUnitSecond fromDate:date];

NSLog(@"%ld, %ld, %ld, %ld, %ld, %ld",com.year,com.month,com.day,com.hour,com.minute,com.second);
```