C#程序设计

赵丽丽


C#
基础概论
语言特点
开发环境
运算符与表达式
算术运算符
关系运算符
控制台输入输出
控制台输入输出
Console.ReadLine()
循环结构
面向对象基础
参数传递
循环结构
for循环
while循环
do-while循环
数组与字符串
运算符与表达式
类与对象
Console.ReadLine()
Console.WriteLine()
分支结构
数组与字符串
if语句
switch语句
方法
数组定义
字符串操作
文件读写
继承
多态
异常处理
try-catch语句
文件类型
File类
StreamReader/StreamWriter

目录

CONTENTS


01
基础语法与数据类型

标识符命名规则与关键字体系

01

合法标识符构成规则

标识符由字母、下划线或数字组成，禁止以数字开头，不得包含@、/、-等特殊符号。C#严格区分大小写，命名需具备可读性与语义化特征。

02

系统关键字与入口约束

class、int、if、static、const、virtual、override等关键字不可作为变量名。控制台程序唯一入口固定为static void Main()方法，这是程序执行的起点。

值类型与引用类型的本质差异

值类型的栈内存存储

int占4字节、long占8字节，double、bool、char等值类型直接存储于栈内存，赋值时创建独立数据副本。

01

引用类型的堆内存寻址

string、数组、自定义类及StreamReader/StreamWriter等引用类型，变量保存堆内存地址，赋值仅复制引用。

02

常量定义与初始化强制

const定义编译期常量，值编译时确定。变量使用前必须初始化，未初始化变量禁止参与任何运算。

02

运算符与表达式

运算符优先级与逻辑表达式规范

算术与复合赋值运算符

/为整数除法，%为取模运算，*=、+=等复合赋值运算符简化自更新操作，提升代码简洁度。

优先级层级与区间写法

优先级顺序：！> 算数运算符 > 关系运算 > 逻辑与或 > 赋值。
数学区间必须写为x>min && x<max。

逻辑运算符的短路特性

&&逻辑与、||逻辑或具有短路求值机制，左侧表达式即可决定结果时右侧不再执行。

三目条件表达式

布尔值?真:假的三目运算符提供简洁分支表达，适用于简单二选一场景的赋值操作。

03

控制台交互机制

输入输出方法的功能

Console.Write()输出后不换行，适用于连续格式化输出；Console.WriteLine()输出后自动换行，用于独立信息展示。二者配合实现灵活排版。

输出方法的行为差异

Console.ReadLine()始终返回字符串，需经int.Parse()显式转换为数值。Console.ReadKey()捕获任意按键，常用于暂停程序等待用户确认。

输入读取与类型转换

04

分支控制结构

if语句的语法陷阱与多分支策略

if语句的分号陷阱

if条件后误加分号导致空语句执行，逻辑体脱离条件控制形成严重bug，此为初学者高频错误。

if-else多分支应用

if-else支持递进多分支，适用于闰年判定、素数筛选、数字区间归属等经典算法场景。

switch的常量约束

switch提供多路等值分支，case标签仅允许常量表达式，禁止直接放置变量，源于跳转表编译优化。

05

循环控制结构

for循环的灵活结构与执行边界

三段式结构的省略机制

for循环初始化、条件、迭代三表达式均可省略，但分号分隔符必须保留。条件缺省形成无限循环，需配合break控制终止。

break与continue的控制差异

break立即终止整个循环并跳出；continue仅结束当前迭代，直接进入下一轮条件判断。二者配合实现复杂流程控制。

while与foreach的适用场景辨析

while的先判断特性

while先判断后执行，条件不满足时循环体零次执行，适用于不确定次数的迭代场景。

do-while的至少执行保证

do-while先执行后判断，循环体至少执行一次，适用于菜单确认等需先展示后判断的场景。

foreach的遍历专精

foreach专为遍历设计，适用于数组、字符串及List<T>，要求对象实现IEnumerable接口。

经典循环算法考点

求和运算、奇偶统计、素数判定、数组逆序等算法是循环结构的核心考题，需熟练掌握。

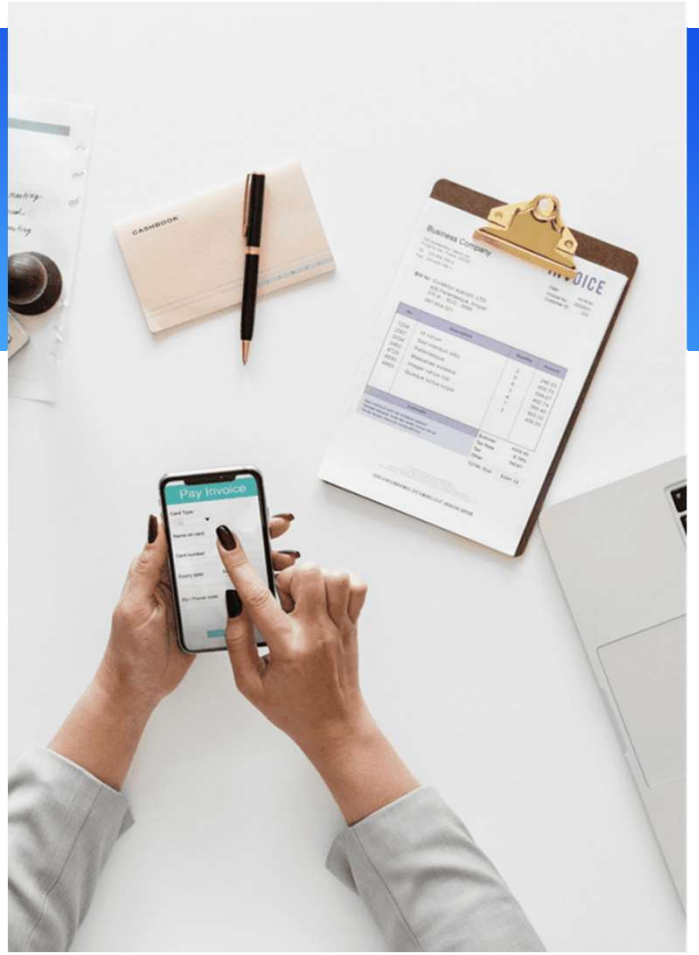
06

数组与字符串操作

一维数组的初始化规则与边界安全

默认填充机制

初始化元素不足时，数值数组默认填充0，string数组默认填充null。理解默认值对避免空引用异常至关重要。



Length属性与索引边界

Length获取元素总数，索引自0起始，最大有效索引为Length-1。循环终止条件必须严格限定以防越界。

数组操作与引用特性

Array.Reverse()实现元素逆序。数组作为引用类型，赋值仅复制引用，修改会影响所有指向同一堆内存的变量。

字符串不可变性与StringBuilder优化

string的不可变本质

string为不可变引用类型，任何拼接操作均生成新对象，导致堆内存频繁分配与垃圾回收压力，Length属性返回字符个数。

StringBuilder的性能优势

StringBuilder采用动态缓冲区机制，支持原地修改，在循环拼接、动态构建等高频场景中效率显著优于string，是高性能字符串操作首选。