自我介绍

2022年3月14日 9:35

姓名：贺志鹏鸟

花名：花生 《最仓》

学号：黑大帅

院校：同济大学

籍贯：湖南

爱好：乒乓球，羽毛球，游泳，跑步..

工作：

规章制度

2022年3月14日 9:41

$6+2=8$ $8\times6=48-2=46km$

学习这件事儿

2022年3月14日 10:10

学习认知：

① 大家来上课是来学习知识的吗？

学习技能！

② 学习使你快乐吗？

学习本身就是有人生的事情。

③ 优秀是一种习惯。

学习方法：

① 提前预习

② 上课认真听讲。

③ 记笔记。（以老师的笔记为主）

④ 当天的作业当天完成。（拓展题目）

就业方向

2022年3月14日 10:50

工程师类别	产品类别	产品形式	主要使用语言
前端开发工程师	浏览器/服务器模式的浏览器端呈现	你看到的所有网页页面	HTML5，JavaScript，CSS 等
	客户端/服务器模式的客户端	手机 APP，QQ，迅雷等各种软件，游戏客户端用游戏引擎，入门可选择 Cocos2d-x，Unity3D 等	Java(安卓)，OC，swift（IOS），C#(windows 端)，C++(游戏前端)
后端开发工程师 主要就业方向	浏览器/服务器模式的服务器端	电商网站的服务器端，知乎，豆瓣	JavaEE(代表有阿里京东)，php，python Go
	客户端/服务器模式的服务器端	游戏，电信，安防，金融类 APP 的服务端	C/Linux 系统编程/C++（腾讯，网易，华为）
	大数据，AI 工程师	做知识图谱，用户画像等数据分析等	Python/C++/Java
嵌入式开发工程师 物联网	运行在非手机，PC，服务器的其他所有设备上	AR，VR，华为基站，大疆无人机，天猫精灵，小米小爱等	C/Linux 系统编程/Linux 内核开发

大数据、AI 方向：推荐、广告、搜索、图像处理、自然语言（AI的掌上明珠）

懂硬件（数电，模电）

Java：设计初衷就是用来开发物联网（Sun）
Web 爆发时期

B/S

C/S → 金融、游戏

Browser：操作简单
传输数据不方便
更新容易

Client：操作复杂
传输数据少
更新比较困难

为什么学习C语言

2022年3月14日 11:13

① 在所有的高级编程语言中，C语言的效率是最高的（Linux）

② 当今的网络世界就是建立在C语言之上的。

操作系统：Linux

高性能服务器：Nginx

缓存中间件：Redis

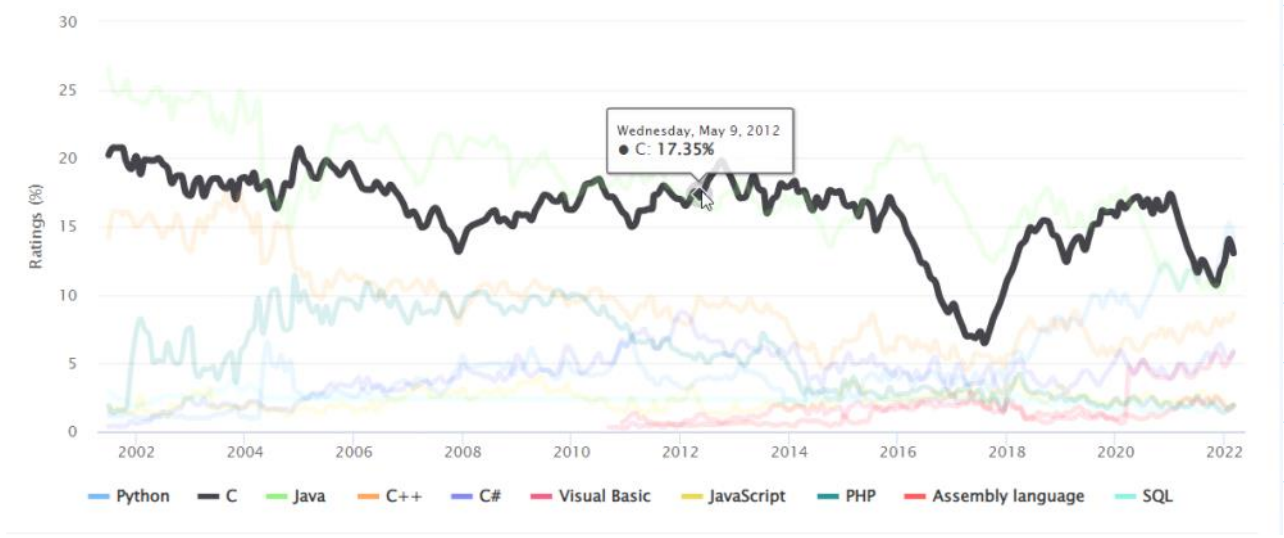
网络协议

硬件驱动

③ C语言影响后来大多数的编程语言：C++ / Java（C++--）/ C# / Go / Rust ..

④ TIOBE指数

Mar 2022	Mar 2021	Change	Programming Language	Ratings	Change
1	3	↑	Python	14.26%	+3.95%
2	1	↓	C	13.06%	-2.27%
3	2	↓	Java	11.19%	+0.74%
4	4		C++	8.66%	+2.14%
5	5		C#	5.92%	+0.95%



C语言的历史

2022年3月14日 11:28

C语言的作者：Dennish Ritch，Ken Thompson （AT&T，Bell实验室）

Ken Thompson：Space Travel （Unix）

BCPL → B → C → 用C语言重写了Unix操作系统

① C语言是一门高级语言，但是它可以操作底层硬件，而且性能非常高。

不会检查数组越界。

没有try--catch机制。

② C语言和汇编语言有比较强的对应关系。

介绍VS的基础操作 (~~****~~)

2022年3月14日 11:42

创建新项目 → 空项目 → 配制新项目

解决方案：管理多个项目

项目1：

头文件：*.h

源文件：*.c

资源文件：配置文件，音频、视频、图片...

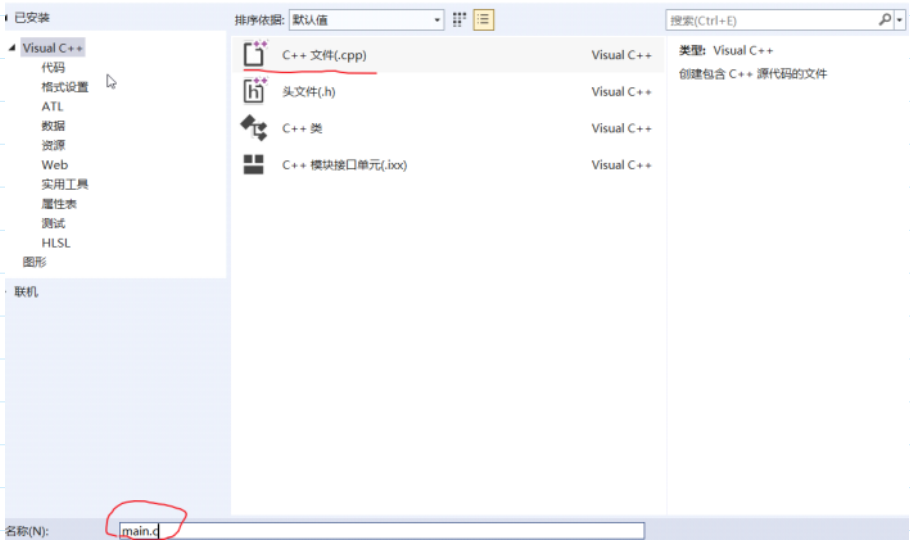
项目2：

头文件

源文件

资源文件

项目3：



错误：

```
1>main.c
1>D:\code\c\43CPP\43C_Day01\1_HelloWorld\main.c(5,7): warning C4013: “prinf”未定义；假设外部返回 int
1>main.obj : error LNK2019: 无法解析的外部符号 _prinf，函数 _main 中引用了该符号
1>D:\code\c\43CPP\43C_Day01\Debug\1_HelloWorld.exe : fatal error LNK1120: 1 个无法解析的外部命令
1>已完成生成项目“1_HelloWorld.vcxproj”的操作 - 失败。
========== 生成: 成功 0 个，失败 1 个，最新 0 个，跳过 0 个 ==========
```

Link 链接时发生的错误：函数名错了，没有它含对应的头文件.

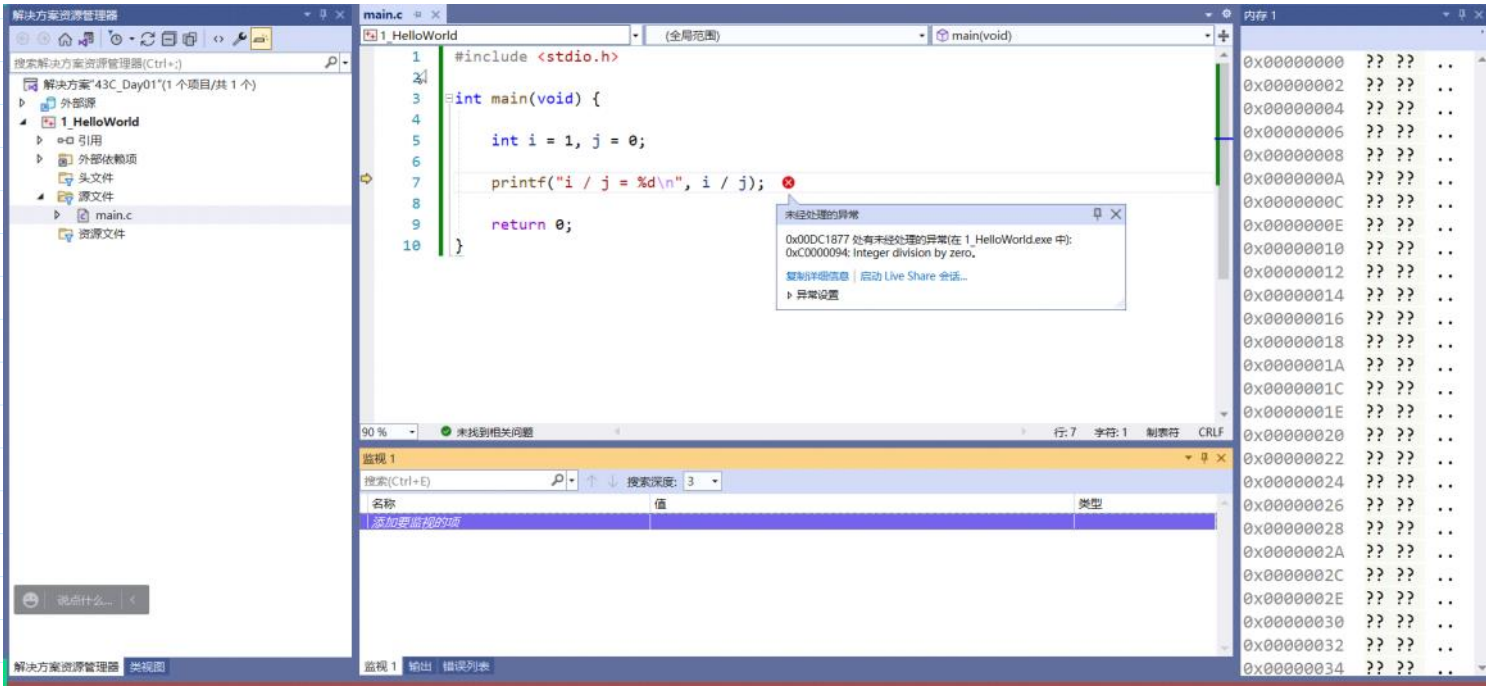
```
1>main.c
1>D:\code\c\43CPP\43C_Day01\1_HelloWorld\main.c(5,4): error C2065: “i”: 未声明的标识符
1>已完成生成项目“1_HelloWorld.vcxproj”的操作 - 失败。
```

Compile 编译发生了错误

```c
int main(void) {
    int i = 1, j = 0;
    printf("i / j = %d\n", i / j);
    return 0;
}
```

运行时错误

调试界面：窗口布局.



调试程序：

打断点（关键点）

F5, F10, F11, Shift + F11, 继续

运行到下一个断点，如果后面没有断点，执行完整个程序.

HelloWorld

2022年3月14日 14:30

返回值类型

```c
#include <stdio.h>
int main(void) {
    printf("Hello world.\n");

    return 0;
}
```

#include <stdio.h> ← 预处理指令，头文件包含

没形参数

printf("Hello world.\n"); ↳ 转义序列，换行

main：操作会调用main函数，程入的入口

问题，0会返回给谁？

操作系统。状态码，0代表正常退出。

非零代表异常退出

内存大小单位：

bit：b

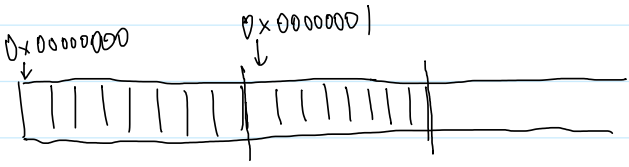
byte：8 bits ⇒ 最小的寻址单位

kb：2^{10} byte

MB：2^{20} byte

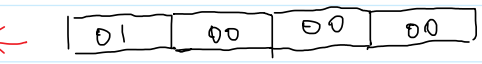
GB：2^{30} byte

TB：2^{40} byte



0x0117F908 01 00 00 00 ⇒ i=1，int类型一般占用4个字节

小端表示法 ←



地址小 ⟶ 大

← 内存布局

大小端的问题

00 00 00 01

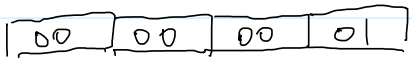
← 机器码

编码

← 真值

小端表示法：低地址存放低有效位

大端表示法：高地址存放低有效位

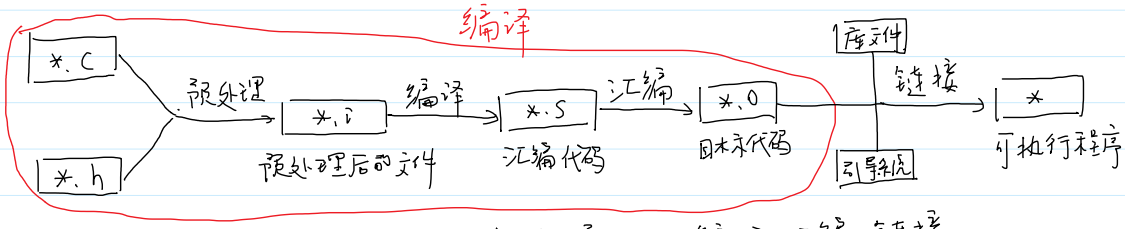


→ 大端表示法

地址小 ⟶ 大

如何生成程序（xxxxx）

2022年3月14日 15:14



生成：预处理、编译、汇编、链接

#1. 预处理

执行预处理指令，以#开头的命令

#include 头文件包含

原理：将对应头文件的内容 copy 到对应的位置

#define N 5 宏定义

原理：简单文本替换

（项目 → 属性 → C/C++ → 预处理器 → 预处理到文件）

#define Foo(x) 1+x*x 带参数的宏（宏函数）

注意事项：① 左括号应该紧贴宏函数的名称

② 把整个宏函数表达式用括号括起来 Foo(x) 1+x+x*x | 3*Foo(5)

③ 把宏函数的参数添加括号 Foo(x) (1+x+x*x) | Foo(1+2)

④ 避免宏函数导致的多次副作用 Foo(x) (1+(x)+(x)*(x)) | Foo(++i)

⑤ 定义完备的多语句宏函数 do...while(1) Foo() do{printf("Hello ");
printf("world.\n");
}while(0)

思考一个问题 既然使用宏函数这么麻烦，为什么C语言还提供它呢？

效率高！

普通函数调用会有一些额外的开销：

调用函数：保存寄存器值，传递参数，保存下一条指令的地址...

函数返回：传递返回值，恢复寄存器值的值...

#2. 编译

把C语言源代码翻译成汇编代码 https://godbolt.org/

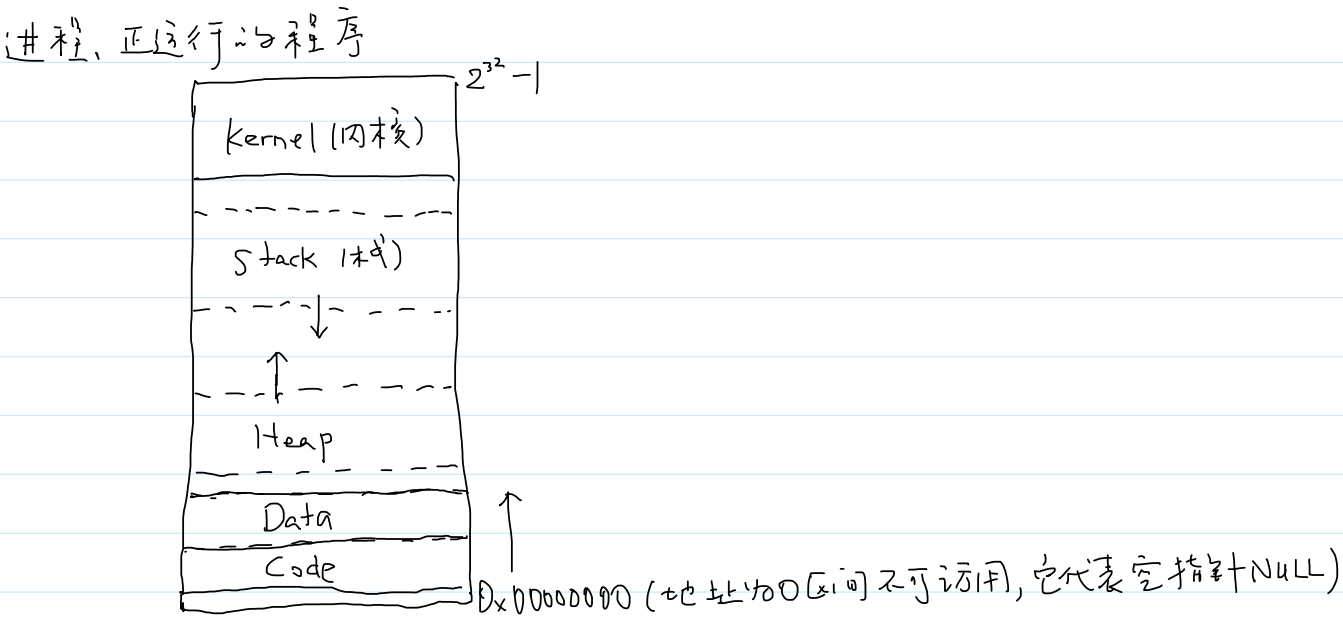
```c
#include <stdio.h>

int main(void) {
    printf("Hello world.\n");
    return 0;
}
```

```
.LC0:
        .string "Hello world."
main:
        push    rbp
        mov     rbp, rsp
        mov     edi, OFFSET FLAT:.LC0
        call    puts
        mov     eax, 0
        pop     rbp
        ret
```

#3. 汇编和链接

进程的虚拟内存空间（[illegible]）

2022年3月14日 16:20

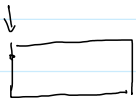


变量和常量

2022年3月14日 16:29

1. 变量：在程序的运行期间可发生改变的量。

变量的本质：一块内存空间



变量的三要素：① 如何引用这块内存区域 → 变量名。

② 这块内存区域有多大？以及我们该如何解释这片内存区域的数据？ → 变量类型

③ 这片内存区域表示怎样的数据 → 值。

int a = 10;

2. 常量

常量：在程序运行期间不可以发生改变的量。

#define N 5

const int M = 5;

常量和常量表达式之间的区别？

常量表达式：在编译期可以直接求值的表达式。(constexpr)

#define N 5 → 常量表达式。

const int m = 5; ⟷ 常量，不是常量表达式。

常量表达式可以用来指定数组的长度，还可以用于switch语句的标签。

```c
#define N 5

const int M = 10;

int main(void) {

    int arr1[N];
    int arr2[M]; // M不是常量表达式

    int i;
    scanf("%d", &i);

    switch (i) {
        case N:
            printf("N = %d\n", N);
            break;
        case M: // M 不是常量表达式
            printf("M = %d\n", M);
            break;
    }

    return 0;
}
```

const int *p;

从右往左看

pointer to const int

int* const p;

从右往左看

constant pointer

标识符和关键字

2022年3月14日 17:10

标识符（identifier）：为变量、宏、函数等起的名字

规则（rules）：① 标识符中只能包含字母、数字和下划线.

② 不能以数字开头.

规范（conventions）

单词与单词以下划线分割 symbal_table, current_page

驼峰命名法：symbalTable, CurrentPage （C++, Java, C#.... ）

好的标识符，见名知义.

注意事项：① 标识符是区分大小写的，Job, Job, JOB

② 标识符不能与关键字冲突.

关键字：对C语言有特殊意义的名字

auto	float	signed	_Alignas (since C11)
break	for	sizeof	_Alignof (since C11)
case	goto	static	_Atomic (since C11)
char	if	struct	_Bool (since C99)
const	inline (since C99)	switch	_Complex (since C99)
continue	int	typedef	_Decimal128 (since C23)
default	long	union	_Decimal32 (since C23)
do	register	unsigned	_Decimal64 (since C23)
double	restrict (since C99)	void	_Generic (since C11)
else	return	volatile	_Imaginary (since C99)
enum	short	while	_Noreturn (since C11)
extern			_Static_assert (since C11)
			_Thread_local (since C11)

在大多数IDE中，关键字会以特殊颜色进行标识

格式化输入和输出

2022年3月14日 17:24

计算机包含哪几个组成部分？

冯诺依曼体系结构（存储程序型计算机）：输入设备、运算器、控制器、存储器、输出设备。

最简单的计算机：输入设备、运算器、输出设备。

#1. 输入和输出的模型 (☒☒☒)

核心矛盾：CPU、内存、I/O设备在处理数据存在速度差异。

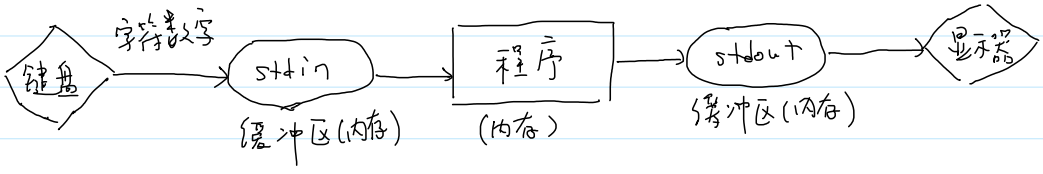
比喻：CPU一天，内存一年（10^2）

内存一天，I/O十年（10^3）

木桶理论：一个木桶能装多少水，取决于最短的木板。

CPU、内存：高速缓存（Cache）

内存、I/O设备：缓冲区（buffer），缓存（Redis）



#2. 格式化输出（printf）

↳ format

函数的原型：int printf(格式化字符串, 表达式1, 表达式2, ...)

作用：显示格式串中的内容，并且在该字符串指定的位置插入要显示的值。

格式化字符串：① 普通字符 → 直接输出

② 转换说明 → 以%开头的，表示一个占位符，我们会以后面表达式的值替换占位符

%d：以十进制形式输出整数

decimal

%f：输出浮点数。

float

```
int i = 9527;
float f = 3.14f;

printf("i = %d, f = %f\n", i, f);
i = 9527, f = 3.140000
```

%d：以整数形式解释该片内存空间，并以十进制形式输出。

%f：以浮点数的形式解释该片内存空间，并输出。

转换说明：① 以何种方式解释内存区域（编码）

② 以何种格式输出。