
七院创新基地文档

发布 1.0

SevenInnovationBase

2020 年 03 月 01 日

1	数学与计算科学学院创新实践基地简介	1
1.1	了解我们	1
1.2	数学与计算科学学院创新实践基地管理制度	2
2	基地公告	3
2.1	数学与计算科学学院第二届程序设计竞赛	3
2.2	第一期假期试炼	5
3	各部门简介	11
3.1	算法研发部	11
3.2	应用软件研发部	12
3.3	大数据技术与分析部	13
3.4	信息安全部	15
4	金句良言	17
5	内部文章	19
5.1	StringBuilder 的高性能使用方式总结 author: CkaiGrac	19
5.2	win10 自带虚拟机 Hyper-V	22
5.3	进制转换题目解析	35
5.4	几种常见的博客搭建资源汇总	39
6	干货推荐	45
6.1	程序员能力矩阵	45
6.2	科学上网	45
6.3	C/C++ 相关	46
6.4	跟我一起写 Makefile	46
6.5	RSA 算法原理	46
6.6	UI/UX	47

6.7	Git 相关	47
6.8	More	47
7	常识须知	49
7.1	入门必知	49
7.2	信息安全相关	50
7.3	大数据技术相关	50
7.4	软件开发相关	50
7.5	前端相关	51
8	文档渲染测试	53
8.1	Markdown 语法渲染测试	53
8.2	rst 渲染测试	56
9	关于本文档	59
9.1	参考资源	59

数学与计算科学学院创新实践基地简介

创新基地，是在原来学院科协基础上建立起来得科技组织，旨在提高大学生的动手实践能力，培养一批具有高素质、创新性的学生团队，为学院科技创新提供一个崭新的平台。

创新基地采用以“项目”为核心，以个人实践为脊梁，以“老师指导”为源泉的运作管模式，有学生自主管理，负责学院的科技创新、动手实践能力、团队创作、团队合作意识的培养以及学院科技项目的申报、实践和验收工作。

1.1 了解我们

如果你想加入我们，那么你需要知道以下几点

1. 创新基地虽然是一个组织，更是一个大家庭，创新基地里面的学长学姐，即使技术超群，牛人一个，他们也不会以一副高傲的样子对待你。相反，他们很热情，对学弟学妹提出的问题都很有耐心的讲解。
2. 创新基地是一个提升自我的平台，面对未来感到迷茫的你，或许会从这个大家庭里面找到一丝丝灵感和方向，如果你肯下功夫肯努力，将大大有利于你未来的就业。
3. 创新基地可利用资源丰富，配备有台式电脑，以及关于编程方面的大部分书籍，不来好好利用一下，你就亏了。
4. 我们七院创新基地，是属于桂林电子科技大学下的科技组织，所以招新对象仅限于桂林电子科技大学数学与计算科学学院的同学。当然，非常欢迎其他学院的同学和我们交流学习编程技术，共同进步。
5. 基地冬暖夏凉哦 ~~~~

1.2 数学与计算科学学院创新实践基地管理制度

1. 指导老师根据成员学习情况分配学习任务，成员应完成本部门相应指导老师规定的学习任务。
2. 由各部门部长定期组织本部门学习心得分享。
3. 由创新创业基地负责人至少每两个月组织开展 1 次创新沙龙。
4. 每学期期末定向对创新创业基地成员进行考核，考核内容包括指导老师布置的学习（项目）任务，平时参加会议出勤情况等。
5. 严禁将创新创业基地电脑带出基地，严禁擅自拆卸使用他人电脑。
6. 最后离开创新创业基地者应当注意关好门窗、切断电源。

2.1 数学与计算科学学院第二届程序设计竞赛

2.1.1 一、活动目的

为提高学院科技创新活动氛围，推动学院科技创新活动的发展，引导同学们学习掌握计算机与互联网知识，提高同学们将所学理论转化为实际应用的能力，同时为了给同学们提供一个了解并参与程序设计竞赛的平台，数学与计算科学学院创新创业实践基地将面向学院全体学生举办“数学与计算科学学院第二届程序设计竞赛”。

2.1.2 二、主办单位与负责老师

- 主办单位：数学与计算科学学院创新创业实践基地
- 学院负责老师：毛睿联系方式：15677359715

2.1.3 三、参赛形式

- 参赛对象：数学与计算科学学院全体学生
- 本次比赛为个人赛，主要考察同学们的编程能力（数据结构和算法基础能力）

2.1.4 四、竞赛环境

- 竞赛语言：C、C++、Java、Python
- 开发环境：Dev-C++、IDLE、Eclipse
- 评判工具：使用开源 OJ 搭建的七院在线评判系统
- 比赛机器操作系统：Windows 7

2.1.5 五、竞赛纪律

- 1、参赛选手严禁携带任何纸质资料
- 2、参赛选手不得携带任何通讯工具，不能携带任何电子产品
- 3、比赛严禁作弊，如若发现代码雷同，立即取消参赛资格

2.1.6 六、竞赛规则

- 参赛时间：11 月 18 日 19: 30——21: 30 (06406)，时长 2 小时
- 比赛题目：问题来自数论、图论、动态规划、数据结构、概率论、计算几何、搜索、贪心、模拟等
- 排名方式：按照解题得分、解题数、解题正确率进行排名
- 参赛选手可能收到的 OJ 反馈信息包括：
 - Pending & Judging : 您的解答将很快被测评，请等待结果
 - Compile Error : 无法编译源代码，点击链接查看编译器的输出
 - Accepted : 解题方法是正确的
 - Wrong Answer : 程序输出结果与判题程序的答案不符。
 - Runtime Error : 程序异常终止

- Time Limit Exceeded : 程序使用的 CPU 时间已超出限制
- Memory Limit Exceeded : 程序实际使用的内存已超出限制

2.1.7 七、奖励设置

- 1、一等奖、二等奖、三等奖：
 - 按照有效参赛选手（至少提交过一道题目）的 10%、20%、30% 颁发奖状和礼品
- 2、最佳“新生”奖
 - 排名最高且解决的问题数大于一道的大一新生，颁发奖状

2.1.8 八、报名方式

- 1、填写在线报名表：
 - https://docs.qq.com/form/fill/DYm5oQ2FiZIVOZWpi?_w_tencentdocx_form=1
- 2、比赛交流 QQ 群：924770329
- 3、报名时间段：2019 年 11 月 3 日——2019 年 11 月 17 日

2.1.9 九、备注说明

- 1、本次比赛最终解释权归数学与计算科学学院创新创业实践基地所有。
- 2、颁奖时间地点赛后通知

2.2 第一期假期试炼

2.2.1 前言

考完试了，是不是贼爽？☺放假了学习也不能忘啊，我就不信你们天天玩游戏或者出去旅游等等 xxx。

2019 年秋季学期我们没有较大的考核也没有签到制度，就算是没有这些，仍有一部分同学坚持来基地学习，当然整天来这里吹水的不算学习。（接下来开始贩卖焦虑，(^_^)）

对于大二的同学，你们应该继续深入你们所感兴趣的领域，坚定一个领域学 3、4 年。不会亏!!! 对于那部分学习方向不明确的同学，偷偷告诉你们，你们和其他人的差距正在进一步拉大（或许你没有真正认识到。为什么？因为在这里你们，你们没有真正的“老师”带着你一步一个脚印的学，基本上全靠的是你自己。同时，你们可能自认为我在广泛的涉猎知识，我也在进步啊！其实不然，你们这个阶段所接触的东西太过粗浅，相比于那部分深入领域学习的同学，你们没有真正碰上有难度的挑战，思想压根得不到有力磨练，而且你们能真正动手实践的地方真滴少。希望你们能明确方向，不要乱动。多多动手实践，引用耗子叔的一句话：有些东西你需要亲身经历和体会才知道，而不是光听别人怎么说。

对于大一的同学, 你们能够来基地自主学习的同学真的很少, 也有可能你们在图书馆或者教室学, 宿舍这种地方就不要“待”了。如果你们了解过其他院和你们同一级的, 希望你们能认识到你们和其他院同一级比差距很大。加把劲, 趁着假期, 好好想想你们以后想做什么, 希望你们大一的时候就能够明确方向, 其他科协也有很多学习方向飘摇不定的同学, 你确定好方向, 拉近差距甚至甩开他们不是三两下。希望下学期, 你们有个固定来基地学习的时段, 哪怕每天一两个小时, 长期积累下来, 那也会有质的飞跃。

前言的最后, 偷偷告诉你们, 下学期, 你们可能会很痛苦, 我们会有两次考核。**第一次**为开学第一个月 (估计 2 月 29 日), **第二次**大概在 5 月初。第一次的主要形式如下:

- 学习总结 PPT 讲解, 可以放上你在做的东西 (大一)
- 项目展示 < 项目展示可以是正在做的, 但是还没有完成, 部分功能可展示 >, 开发时注意解耦, 松耦合 (大二)
- 某个知识点的深入理解和实践的分享, 比如 SQL 优化实践, 优化前后效率对比、Git 原理剖析、算法模型讲解 (实现与应用)、TDD (Test-driven development)、GitHub Actions 应用、危险的指针 (the dangerous pointer) 等等, **PPT+ 代码片段** (大二)

第二次估计为项目展示 (成品), 到时候再说 ☺

By the way, 还没有加 GitHub 组织的同学, 在这个 [Issue](#) 里留言, 留下 GitHub 注册邮箱, owner 会发邀请加入的邮件给你们!!!

issue: <https://github.com/seven-innovation-base/SphinxDOC/issues/7>

2.2.2 正题

非正题扯完了, 说正事。这个假期, 我们玩点有意思的, 主题名为: “**AR1SR2W1W2**”。

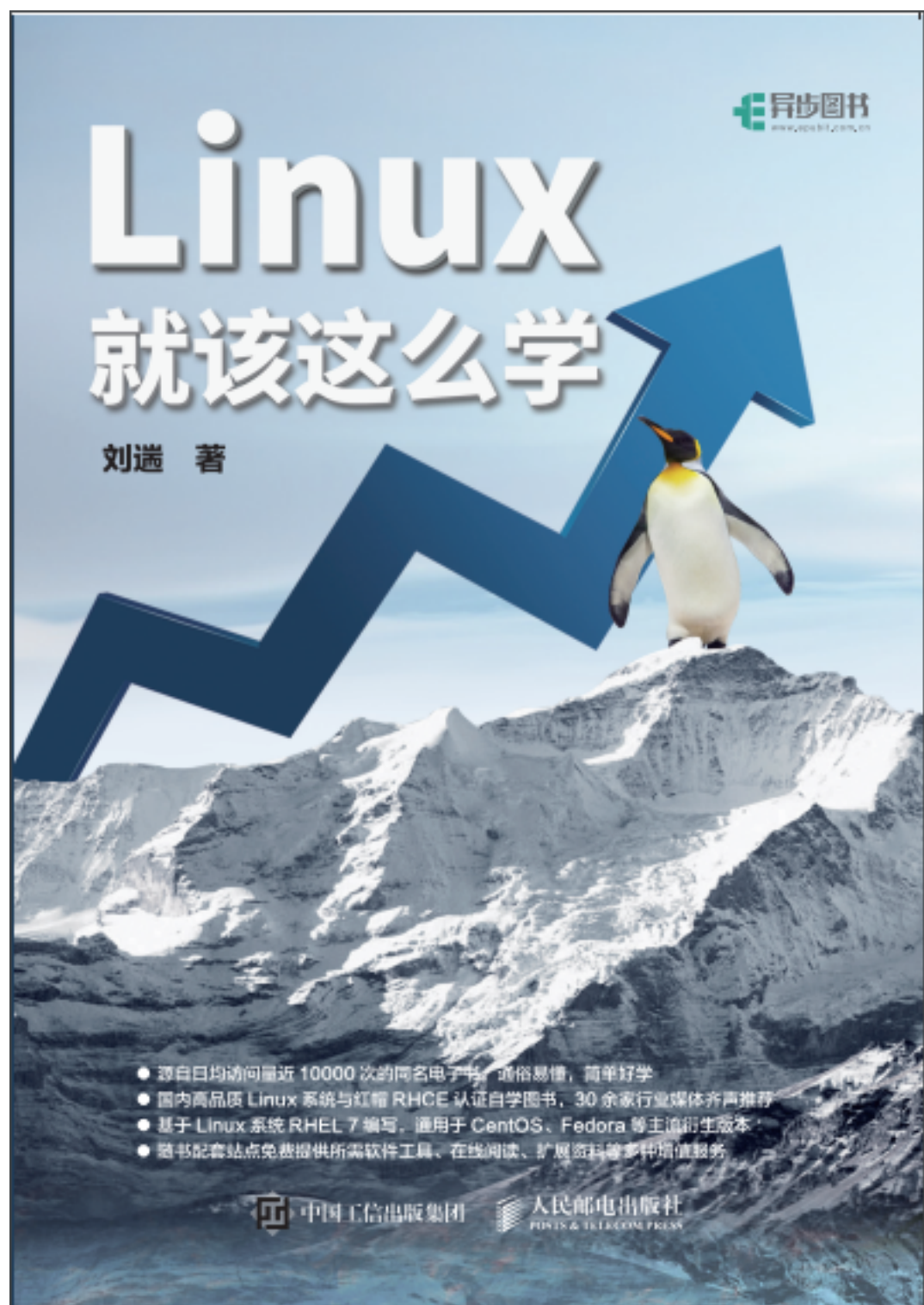
- A: 算法 (Algorithms), 做一道中等难度及以上的算法题或者学习一个有难度的算法, 写下你的解题思路 (多个)、想法、算法原理, 把文章分享出来, 文章可发表在各大技术平台或者你的个人博客、GitHub Project README
- R1: 复审, 检查 (Review), Review 一篇英文技术文章, 把它翻译出来 (跑通作者的代码, 附带你的个人想法), 然后把你整的以文章的形式记录分享出来
- S: 分享 (Share), 写一篇文章, 分享你进入大学到现在的学习感悟或者分享你的技术学习路线, 脑图 + 白话文
- R2: 阅读 (Read), 读一篇有深度的技术文章或者读一本科普书 (比如人类简史), 写下你的感悟, 同样的把你的感悟以文章的形式分享出来
- W1: 看 (Watch), 看一场技术主题演讲的视频, 认识下大佬, 以文章的形式写下你的感悟分享出来。不知道有哪些开发者技术活动, 可以看看这篇文章: [《权威发布 | SegmentFault 思否 2019 中国最受开发者欢迎的技术活动》](#), 这些活动的视频和 PPT 大部分都会放出来, GitHub、B 站或者官网应该能找到
- W2: 写 (Write), 写篇技术文章分享出来, 主题随意, 你们懂得

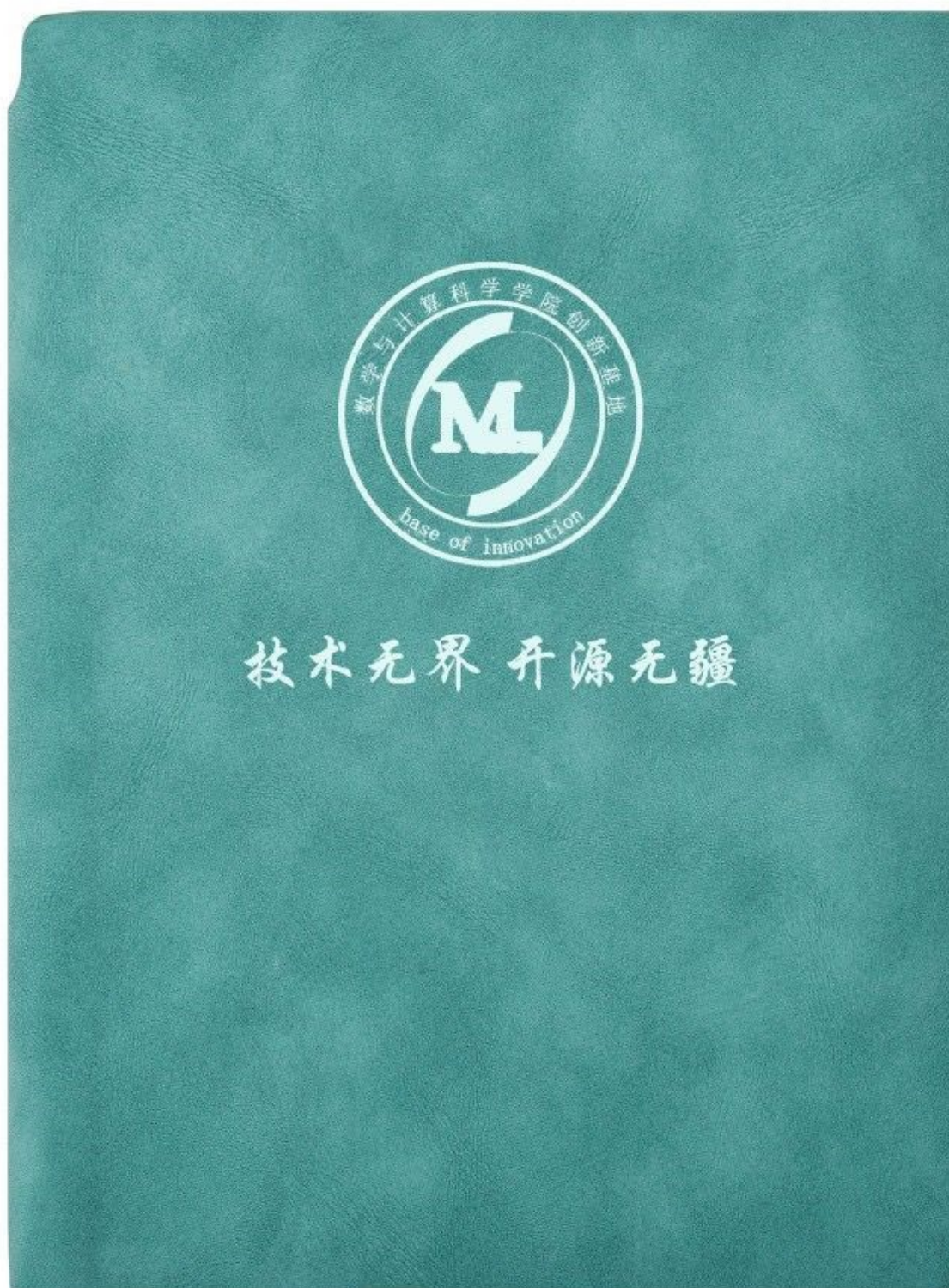
以上, 随便选个两三个或者全部, 我会邀请学长学姐给你们打分。拒绝 Copy、Paste。我们知道, 网络的力量是巨大的, 文字的力量也是巨大的, 要对你们自己写的东西负责哦 (๖o๖)。📄 请大家将文章提交到这里:
<https://github.com/seven-innovation-base/articles/issues/1>

活动时间段为: 2020 年 1 月 1 日-2020 年 2 月 29 日

问题来了, 这么骚, 有没有奖励, 有的喂!!! 请看下面:

- 一等奖: 《Linux 就该这么学》纸质书一本 + 有基地 logo 的商务记事本一本
- 二等奖: U 盘 + 有基地 logo 的商务记事本一本
- 三等奖: 带有基地 logo 的商务记事本







2.2.3 结语

- 最后，祝大家 2020 快乐，诚挚的邀请你们加入本文档的编辑与维护工作，详情请阅读文档项目的[README](#)
- 如果还不会 Git 的话，到练习仓试手 [Git2Github-practice](#)，Git 是现阶段必修的一门技术，多了解下 GitHub，开源正经历着前所未有的繁荣，当你迈出第一步的时候，你就比那些踌躇不定的同学强了

3.1 算法研发部

3.1.1 简介

部门以培养成员扎实的算法能力和编程思想为目标，提升每个成员的算法优化能力、逻辑思维能力和抽象建模能力，为今后的算法比赛和就业做准备。

- 我们学什么? (๑• ̎• ̎)
- C/C++
- C++ STL
- 基础算法
 - * 排序：快排、归并、计数
 - * 搜索：回溯、递归、剪枝
 - * 图论：最短路径、最小生成树、网络流建模
 - * 动态规划：背包问题、最长子序列、计数问题
 - * 基础技巧
 - 分治
 - 倍增
 - 二分

· 贪心

– 数据结构

- * 数组与链表
- * 栈与队列
- * 树与图
- * 哈希表
- * 堆：大小根堆、可并堆
- * 字符串：字典树、后缀树

3.2 应用软件开发部

3.2.1 简介

部门以培养计算机技术领域优秀的工程师为目标，学习方向广泛。主要有：Web 开发、移动端开发（Android、iOS、Hybrid）、PC 客户端开发、DevOps、云计算、云原生等等。

3.2.2 对部门成员的希冀

1. 希望各成员能通过理论和**实践**相结合的方式，巩固和拓展学到的知识，打好编程基本功
2. 兴趣为导，\$ 为辅，本科阶段深度耕耘一个技术领域 2-3 年
3. 开源正经历着前所未有，希望各成员积极参与一些优秀的开源项目的文档贡献和代码贡献

Slogan: 只会侃侃而谈却做不出来、做不到就是个屁!!!

3.2.3 技术生态一览

通用：Git-Version Control、GitHub、Basic Terminal Usage、Data Structure & Algorithm、Semantic Versioning、SSH、HTTP/HTTPS and APIs、Character Encodings、Search、Design Patterns

- **Web 方向**

资料参考：<https://github.com/kamranahmedse/developer-roadmap>

- **移动端开发（Android、iOS、Hybrid）**

资料参考：极客时间开源项目 ~ 程序员技能图谱

- **DevOps**

DevOps 元素周期表

PERIODIC TABLE OF DEVOPS TOOLS (V3)

Legend		Legend		Legend		Legend	
Os	Open Source	Source Control Mgmt.	Deployment	Analytics			
Fr	Free	Database Automation	Containers	Monitoring			
Fm	Freemium	Continuous Integration	Release Orchestration	Security			
Pd	Paid	Testing	Cloud	Collaboration			
En	Enterprise	Configuration	AIOps				
1 Os	Gl	GitLab					
3 Fm	Gh	GitHub					
4 En	Dt	Datical					
11 Os	Sv	Subversion					
12 En	Db	DBMaestro					
19 En	Cw	ISPW					
20 En	Dp	Delphi					
21 Os	Jn	Jenkins					
22 Fm	Cs	Codship					
23 Os	Fn	FitNesse					
24 Fr	Ju	JUnit					
25 Fr	Ka	Karma					
26 Fm	Su	SoapUI					
27 En	Ch	Chef					
28 Fr	Tf	Terraform					
29 En	XLd	Xebialabs XL Deploy					
30 En	Ud	UrbanCode Deploy					
31 Os	Ku	Kubernetes					
32 Fm	Cc	CA CD Director					
33 En	Pr	Plutora Release					
34 Pd	Al	Alibaba Cloud					
35 Os	Os	OpenStack					
36 Os	Ps	Prometheus					
37 Pd	At	Artifactory					
38 Fm	Rg	Redgate					
39 Pd	Ba	Bamboo					
40 Fm	Vs	VSTS					
41 Fr	Se	Selenium					
42 Fr	Jm	JMeter					
43 Os	Ja	Jasmine					
44 Pd	Sl	Sauce Labs					
45 En	An	Ansible					
46 Os	Ru	Rudder					
47 En	Oc	Octopus Deploy					
48 Os	Go	GoCD					
49 Os	Ms	Mesos					
50 Pd	Gke	GKE					
51 Fm	Om	OpenMake					
52 Pd	Cp	AWS CodePipeline					
53 Pd	Cy	Cloud Foundry					
54 En	It	ITRS					
55 Pd	Nx	Nexus					
56 Os	Fw	Flower					
57 Os	Tr	Travis CI					
58 Fm	Tc	TeamCity					
59 Os	Ga	Gatling					
60 Fr	Tn	TestNG					
61 Fm	Tt	Tricentis Tosca					
62 Pd	Pe	Perfecto					
63 En	Pu	Puppet					
64 Os	Pa	Packer					
65 Fm	Cd	AWS CodeDeploy					
66 En	Ec	ElectricCloud					
67 Os	Ra	Rancher					
68 Pd	Aks	AKS					
69 Os	Rk	Rkt					
70 Os	Sp	Spinnaker					
71 Pd	Ir	iron.io					
72 Pd	Mg	Moogsoft					
73 Fm	Bb	BitBucket					
74 En	Pf	Perforce					
75 Fm	Cr	Circle CI					
76 Pd	Cb	CodeBuild					
77 Fr	Cu	Cucumber					
78 Os	Mc	Mocha					
79 Os	Lo	Locust.io					
80 En	Mf	Micro Focus UFT					
81 Os	Sa	Salt					
82 Os	Ce	CFEngine					
83 En	Eb	ElasticBox					
84 En	Ca	CA Automate					
85 En	De	Docker Enterprise					
86 Pd	Ae	AWS ECS					
87 Fm	Cf	Codefresh					
88 Os	Hm	Helm					
89 Os	Aw	Apache OpenWhisk					
90 Os	Ls	Logstash					
91 En	XLi	Xebialabs XL Impact					
92 Os	Ki	Kibana					
93 Fm	Nr	New Relic					
94 En	Dt	Dynatrace					
95 En	Dd	Datadog					
96 Fm	Ad	AppDynamics					
97 Os	El	ElasticSearch					
98 Os	Ni	Nagios					
99 Os	Zb	Zabbix					
100 En	Zn	Zenoss					
101 En	Cx	Checkmarx SAST					
102 En	Sg	Signal Sciences					
103 En	Bd	BlackDuck					
104 Os	Sr	SonarQube					
105 Os	Hv	HashiCorp Vault					
106 En	Sw	ServiceNow					
107 Pd	Jr	Jira					
108 Fm	Tl	Trello					
109 Fm	Sk	Slack					
110 Fm	St	Stride					
111 En	Cn	CollabNet VersionOne					
112 En	Ry	Remedy					
113 En	Ac	Agile Central					
114 Pd	Og	OpsGenie					
115 Pd	Pd	Pagerduty					
116 Os	Sn	Snort					
117 Fm	Tw	Twilio					
118 En	Ck	CyberArk					
119 En	Vc	Veracode					
120 En	Ff	Fortify SCA					

图 1: DevOps Tools

来源: <https://xebialabs.com/periodic-table-of-devops-tools/>

- 云原生技术生态

CNCF Cloud Native Definition v1.0

landscape 来源: [cncf/landscape](https://cncf.io/landscape)

3.3 大数据技术与分析部

3.3.1 简介

部门是在统计学成为一级学科及大数据和人工智能新时代的背景下建立起来的。部门以培养优秀的大数据工程师、数据分析师、大数据科学家为目标。打好扎实的数学与统计基础,掌握数据挖掘的基础理论、核心算法、关键技术及软件系统,熟悉海量复杂数据的组织、处理、分析、挖掘和可视化等问题。

- 我们学什么、要了解什么:
 - 编程语言
 - * Python
 - * Java
 - * R

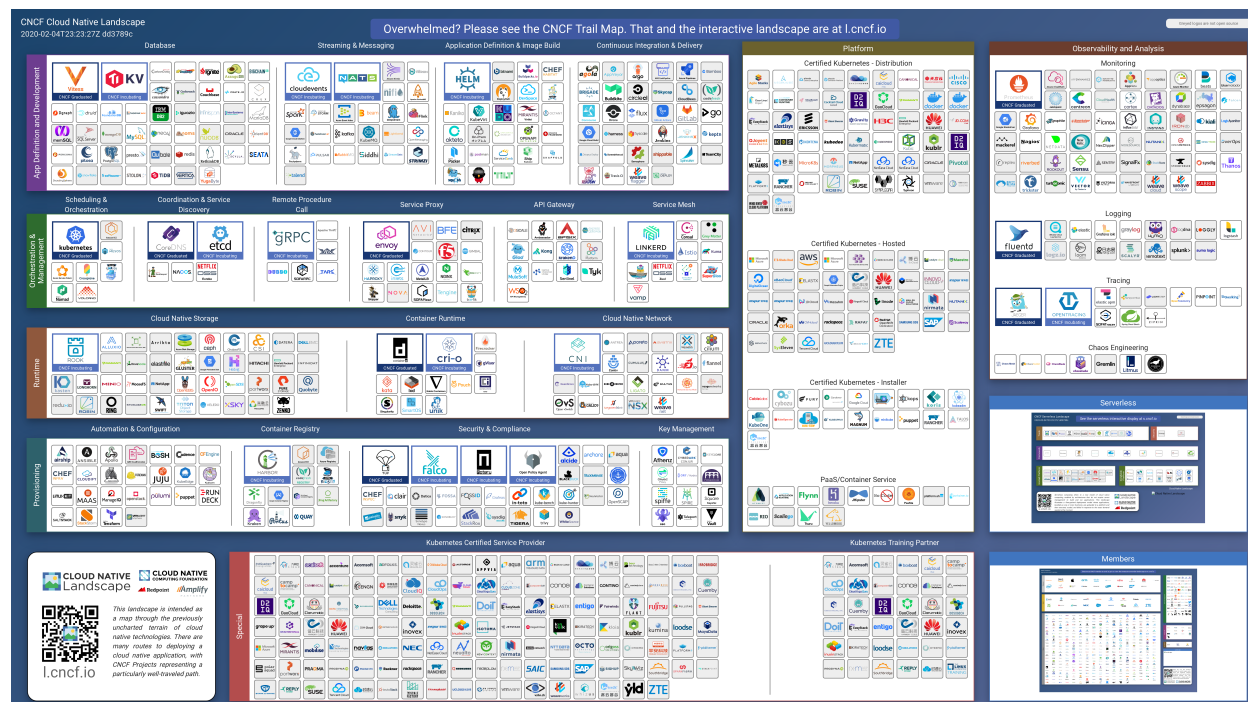


图 2: landscape

- 数据分析挖掘
 - * SPSS
 - * SAS
- 数理统计基础
- 大数据通用处理平台
 - * Spark
 - * Hadoop
- 机器学习基础
- 数据库技术
- 分布式存储、调度技术
- 基础算法

3.4 信息安全部

3.4.1 简介

信息安全部研究范围较广，主要涉及木马编程、网络渗透、逆向工程、代码审计、密码学及社会工程学等方向。部门不同方向的成员可以通过相互组队，参加各大 CTF 夺旗赛及信息安全比赛。

木马编程：木马编程主要学习 C/C++、Windows 编程及 MFC 等，打好编程基础后学习 Socket 编程、DLL 动态调用，尝试去写键盘记录等程序，再添加文件管理、视频监控、音频控制等功能，最后解决自启动、过 UAC、免杀等问题，最终开发出一款成熟的远控。

网络渗透：web 渗透测试需要熟悉网站、数据库、操作系统、计算机网络、脚本编程等方面的知识，主要是针对网站或网络主机服务器群组，利用相关漏洞进行攻击，获取数据，得到 webshell，甚至拿下整个内网。学习渗透测试，要有一定的编程能力，熟练使用各种工具，时刻关注最新漏洞，了解漏洞的基本原理及利用方法，高效率，多实践。

逆向工程：逆向工程主要通过阅读软件的反汇编代码来推断其数据结构、体系结构和程序的设计思路。学习逆向要有良好的编程基础，要精通 C/C++、汇编语言、二进制、算法和数据结构以及各种调试工具。

代码审计：代码审计也叫白盒测试，一般是对网站源码进行分析，从而寻找代码中的 bug。主要学习 PHP、JSP、.NET 等语言以及各种数据库，需要熟悉常见 web 漏洞原理，并且要有较强的逻辑分析能力和代码阅读能力。

3.4.2 职业体系（非正式）

- **Web 安全测试**
 - **Windows：**对网站、业务系统进行安全评估测试(黑盒、白盒测试)；对各类系统进行安全加固；对安全事件进行响应，清理后门，根据日志分析攻击途径…
 - **Android：**负责移动应用安全测试和安全事件应急响应；负责移动应用安全平台建设；参考 Web 安全 Windows
- **Web 安全研发**互联网运营平台、业务安全平台等系统的建设和开发；对安全产品的易用性改进和 Web 研发技术优化；根据需要开发安全辅助工具或平台；参与安全系统的需求分析、设计、编码等开发工作；维护现有的安全程序与系统…
- **病毒分析**病毒样本分析及处理；木马编程…
- **逆向工程**
 - **Windows：**参与漏洞挖掘、分析和利用技术研究；分析高危漏洞原理和利用技巧，撰写相关技术总结文档；挖掘浏览器、Office、Adobe Reader、flash 等客户端软件以及网络协议常见漏洞；研发相应的漏洞挖掘和分析的一体化框架平台；恶意程序分析以及其他程序内部实现逻辑分析…
 - **Android：**开发安卓平台加固框架；参考逆向工程 Windows

非正式划分，如想了解更多详细标准，可以浏览各大招聘网站或相关企业官网的招聘需求

3.4.3 学习方法

拼图法：组织管理学习目标，建立知识技能体系，模块式完成目标。

以 **Web 安全测试** 为例，这是该职业体系的一个基本技能树：BUGBANK-SKILLS > 以目标为导向，一个模块一个模块的完成学习任务，在学习过程中又根据需求自驱动的学习其他相关知识。

3.4.4 学习资源

善于利用搜索引擎，授人以鱼不如授人以渔。

- Google
- Bing
- 知乎
- ...

百度：搜索语法

CHAPTER 4

金句良言

重要：1、多花时间做事情，少花时间说服别人。

2、学习不是为了找到答案，而是为了找到方法。

3、技术广度是技术深度的附属品。

4、选择对了合适的路，比光顾着赶路要重要得多。

5、要应付并通过面试并不难，但是，千万不要应付你的人生。

- 来自陈皓（左耳朵耗子）、CoolShell
-

5.1 StringBuilder 的高性能使用方式总结 author: CkaiGrac

5.1.1 一、设置合理的初始长度

在 `StringBuilder` 的源码中，有一个 `char[]` 数组，这个就是用来存储字符的。

```
//AbstractStringBuilder.java
/**
 * The value is used for character storage.
 */
char[] value;
/**
 * The count is the number of characters used.
 */
int count;
```

而 `count` 统计字符数量。当直接 `new StringBuilder();` 时，传递到父类的默认大小 (capacity) 为 16，也就是默认状态下 `char[]` 数组的长度为 16。来看一下 `append` 方法：

```
//AbstractStringBuilder.java
public AbstractStringBuilder append(String str) {
    if (str == null)
        return appendNull();
```

(下页继续)

(续上页)

```
int len = str.length();
ensureCapacityInternal(count + len);
str.getChars(0, len, value, count);
count += len;
return this;
}

private void ensureCapacityInternal(int minimumCapacity) {
    // overflow-conscious code
    if (minimumCapacity - value.length > 0) {
        value = Arrays.copyOf(value,
                               newCapacity(minimumCapacity));
    }
}

public static char[] copyOf(char[] original, int newLength) {
    char[] copy = new char[newLength];
    System.arraycopy(original, 0, copy, 0,
                     Math.min(original.length, newLength));
    return copy;
}

private int newCapacity(int minCapacity) {
    // overflow-conscious code
    int newCapacity = (value.length << 1) + 2;
    if (newCapacity - minCapacity < 0) {
        newCapacity = minCapacity;
    }
    return (newCapacity <= 0 || MAX_ARRAY_SIZE - newCapacity < 0)
        ? hugeCapacity(minCapacity)
        : newCapacity;
}
```

从上述源码中, 可以看到 `append` 方法接收 `String` 类型的参数, 然后调用 `ensureCapacityInternal()` 方法, 在这个方法里面我们可以看到做了一个判断, 最小容量 `minimumCapacity` 是否比数组的长度大, 而最小容量的大小是 `len+count`。数组长度大于最小容量的话就要扩容, `Arrays.copyOf(value, newCapacity(minimumCapacity));` 这段代码执行数组拷贝。从 `Array.copyOf(char[] original, int newLength)` 可以看到第二个参数是数组新的长度, 而这个新的长度来自 `newCapacity()` 的返回值。在 `newCapacity()` 中我们看到 `int newCapacity = (value.length << 1) + 2;`, `char[]` 数组的长度左移一位再加 2, 也就是长度 $\times 2 + 2$ 。从上面的分析中就可以看到数组的扩容是成倍的, 如果 `StringBuilder` 不设置合适的初始长度就会进行扩容, 造成一点内存损失。

5.1.2 二、重用 StringBuilder

```
public class MyClass{
    private static Field stringBuilderValueField;
    private static Boolean needGetField = true;
    //ThreadLocal 适用于变量在线程之间隔离, 方法之间共享
    private static ThreadLocal<StringBuilder> sbBuffer = new ThreadLocal
    <StringBuilder>();

    //重用 StringBuilder 的例子
    public static StringBuilder obtainStringBuilder(){
        StringBuilder stringBuilder = sbBuffer.get();
        if (stringBuilder == null) {
            System.out.println("stringBuilder 为空");
            stringBuilder = new StringBuilder(1024 * 10);
            sbBuffer.set(stringBuilder);
        } else {
            System.out.println("stringBuilder 不为空");
            stringBuilder.delete(0, stringBuilder.length());
        }
        return stringBuilder;
    }
}

//调用方式为
StringBuilder stringBuilder = MyClass.obtainStringBuilder();
StringBuilder stringBuilder2 = MyClass.obtainStringBuilder();
stringBuilder.append(...);
stringBuilder2.append(...);
```

从上述的代码中可以看到, obtainStringBuilder() 方法返回一个 StringBuilder 对象, 方法中首先从 ThreadLocal 中取 StringBuilder 然后判断是否为空, 如果为空的话则初始化一个 StringBuilder 对象, 大小根据实际情况设置。否则的话就清空 StringBuilder 的内容来重用。对于同一个线程来说, 每次调用 obtainStringBuilder() 方法拿到的都是同一个 StringBuilder 对象。但是对于不同线程来说, 两个线程之间的 StringBuilder 对象是相互独立的, 没有任何关系。这样的好处是减少申请内存的频率, 降低 GC (垃圾回收) 的发生

5.2 win10 自带虚拟机 Hyper-V

author: upsilverfox

5.2.1 教你如何使用 win10 自带虚拟机 Hyper-V

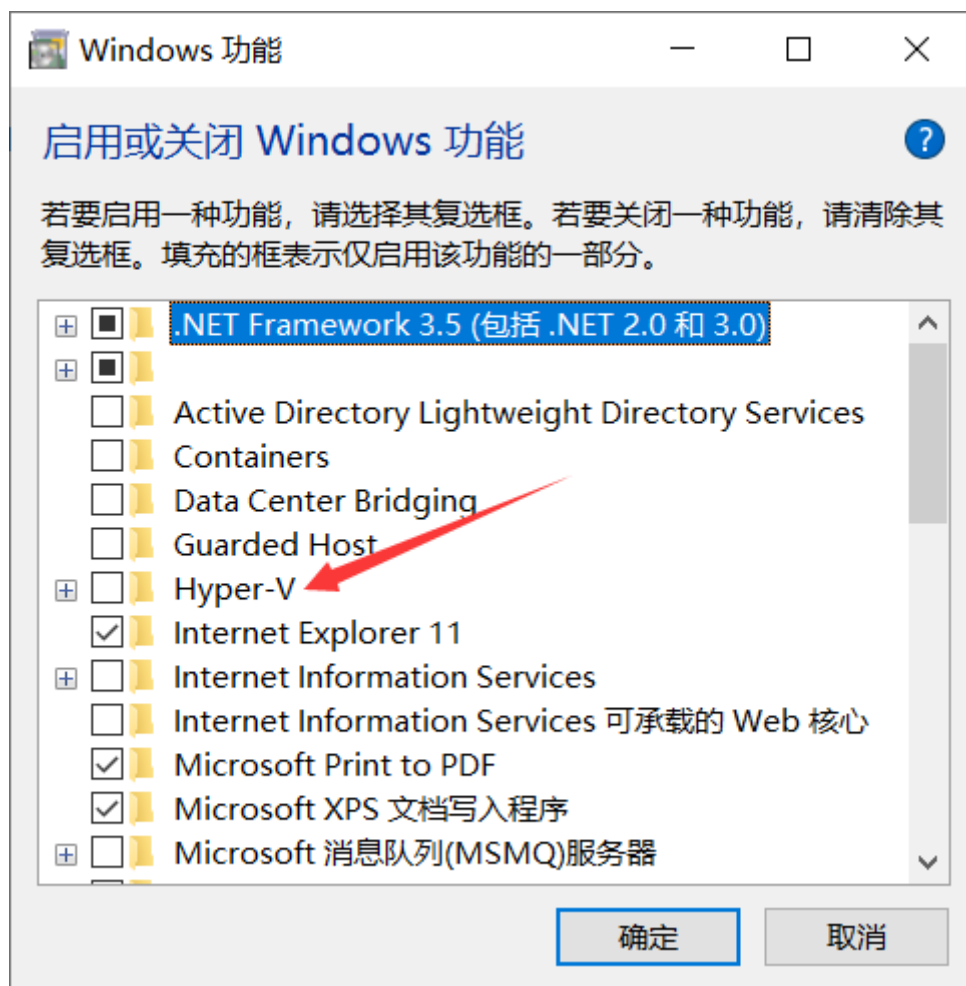
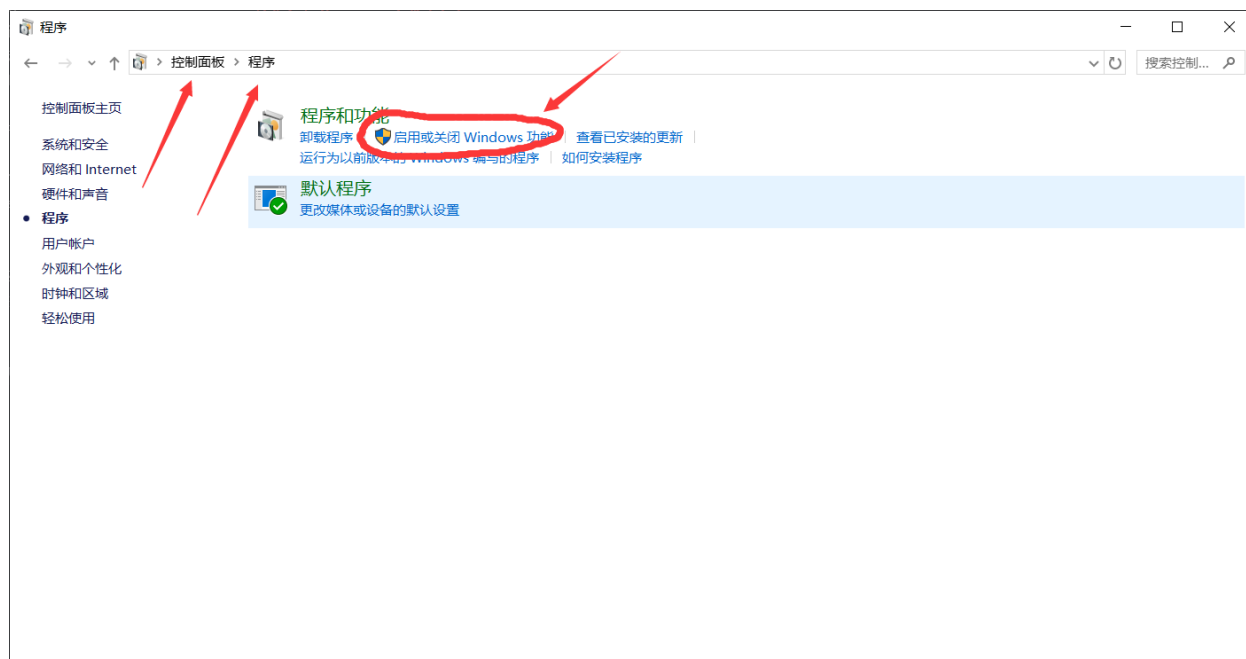
1. 按 win+pause break 快速打开控制面板的系统和安全中的系统，点击控制面板



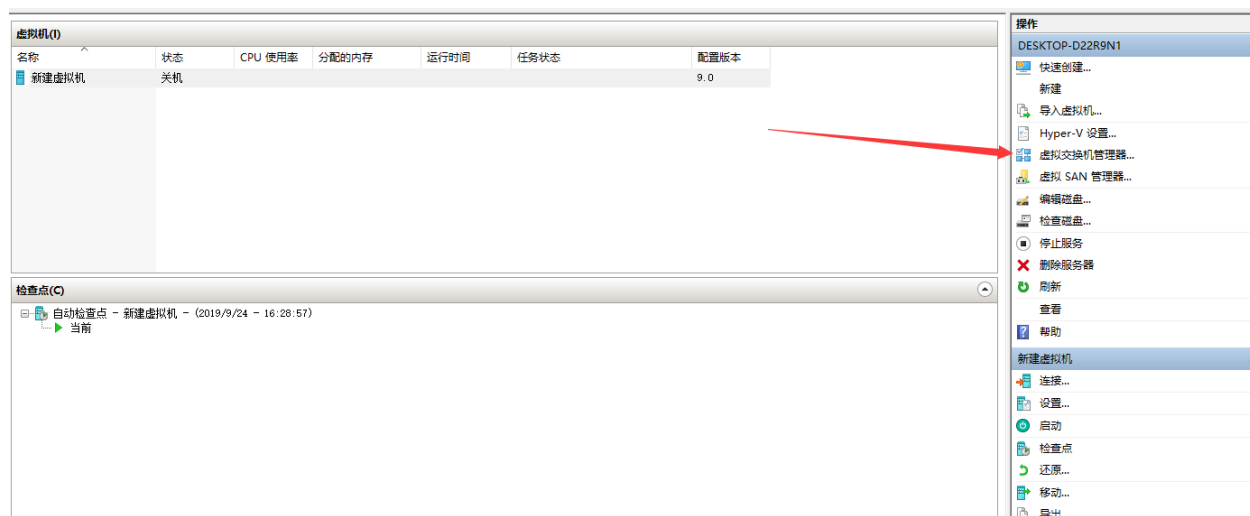
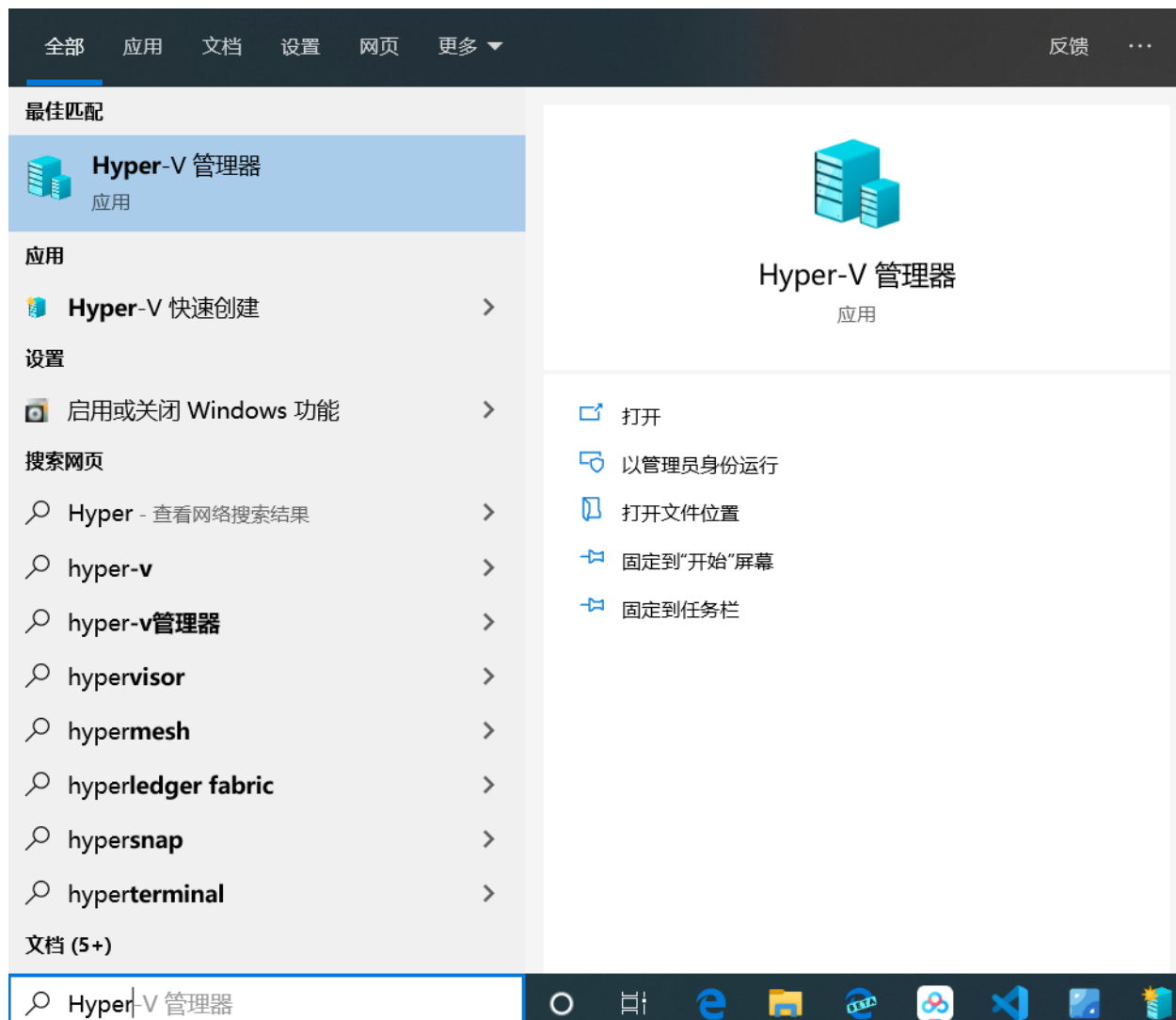
2. 在控制面板中，打开程序组件，选择程序与功能。点击启动或关闭 windows 功能。
3. 找到 Hyper-V，打勾（注意 win10 家庭版时没有 Hyper-V）
4. 勾选后，点击立即重启
5. 开机后找到 Hyper-V，打开程序

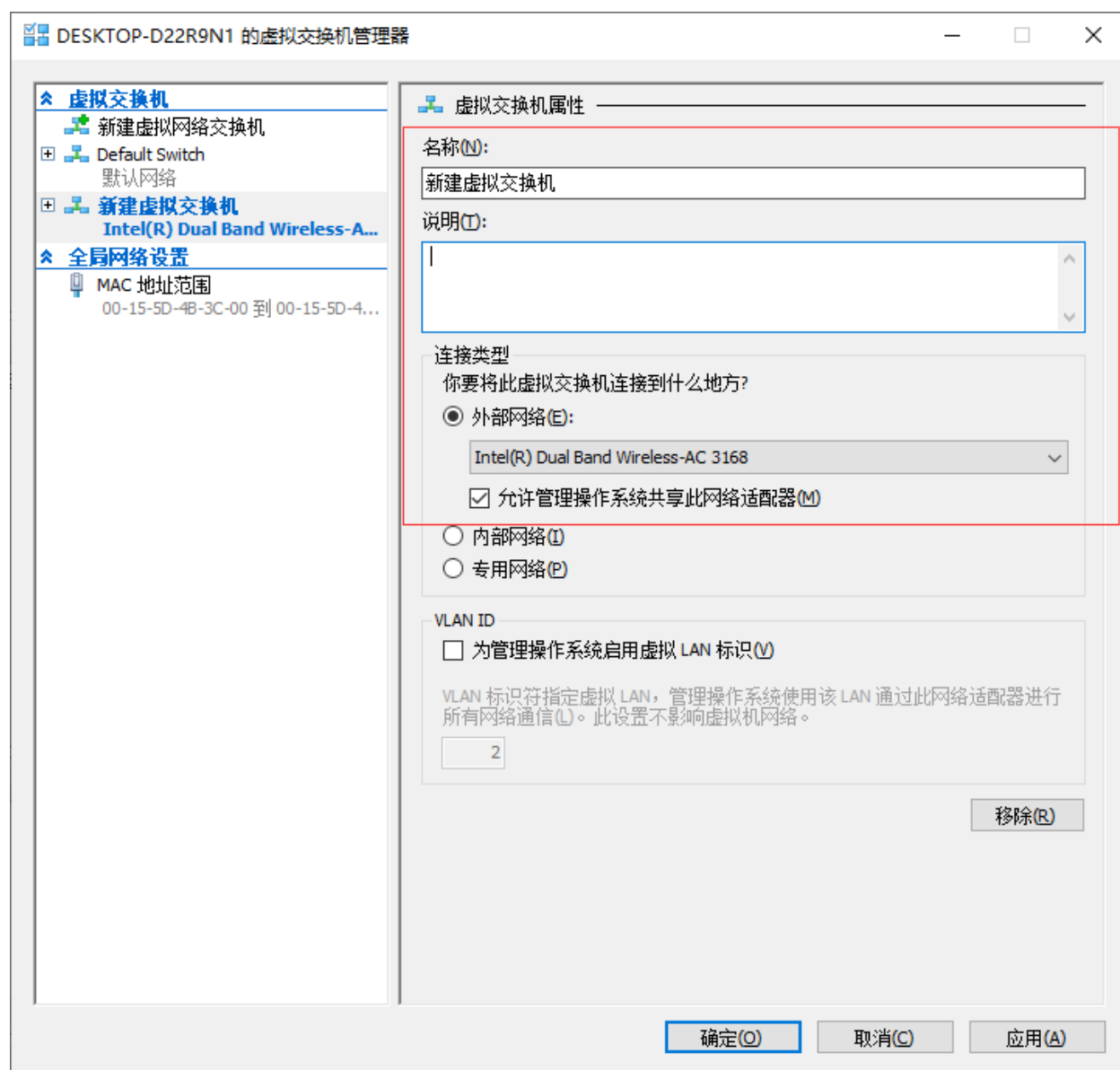
5.2.2 虚拟机网络配置

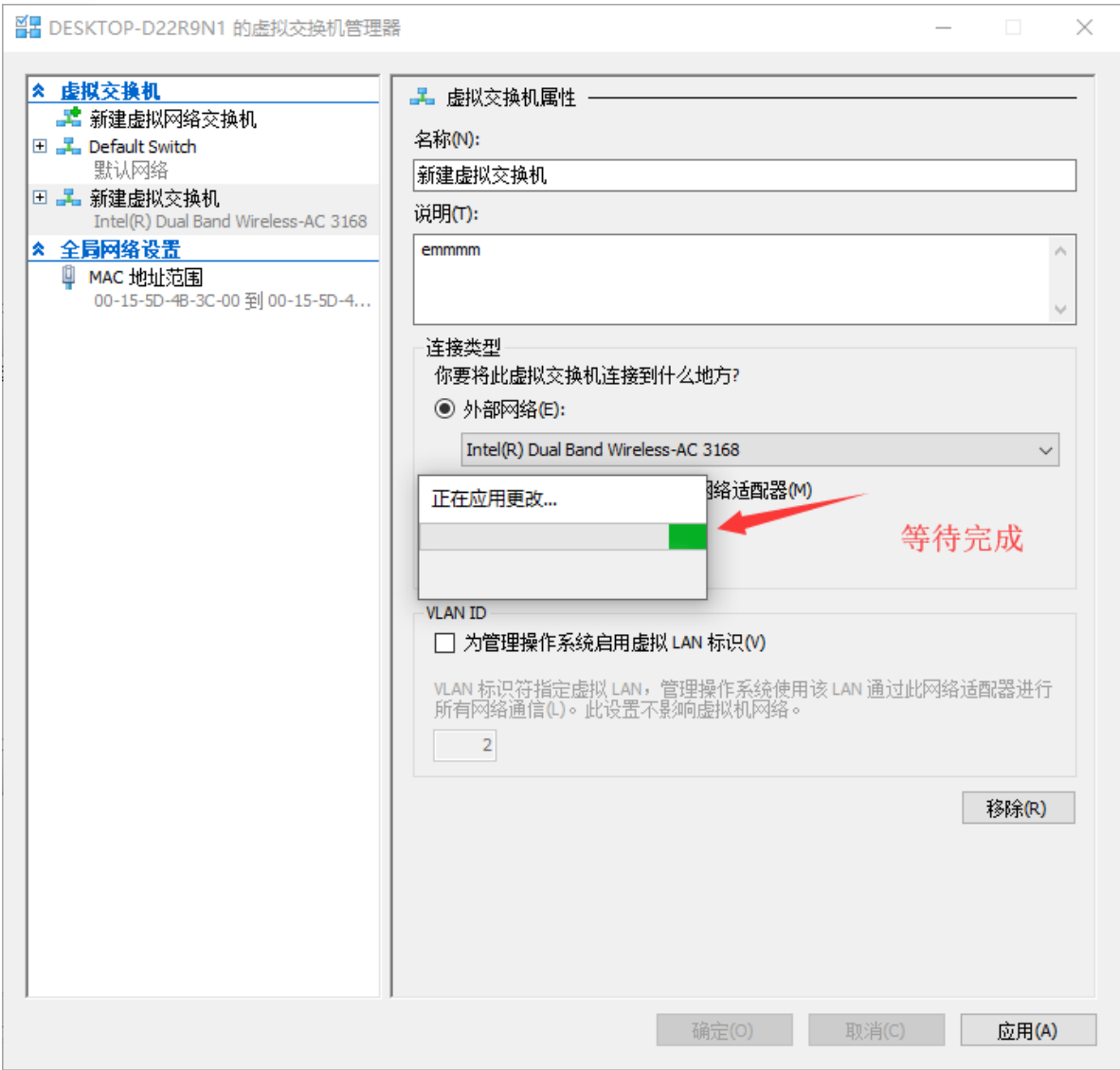
1. 打开 Hyper-V，在右侧操作列表中点击“虚拟交换机管理”，打开虚拟交换机管理器
2. 选择“新建虚拟网络交换机”，在虚拟交换机类型列表中选择“外部”，点击“创建虚拟交换机”，创建一个虚拟交换机，输入虚拟交换机的名称和说明，为虚拟交换机选择一个网络适配器（即物理网卡），点击“确定”。





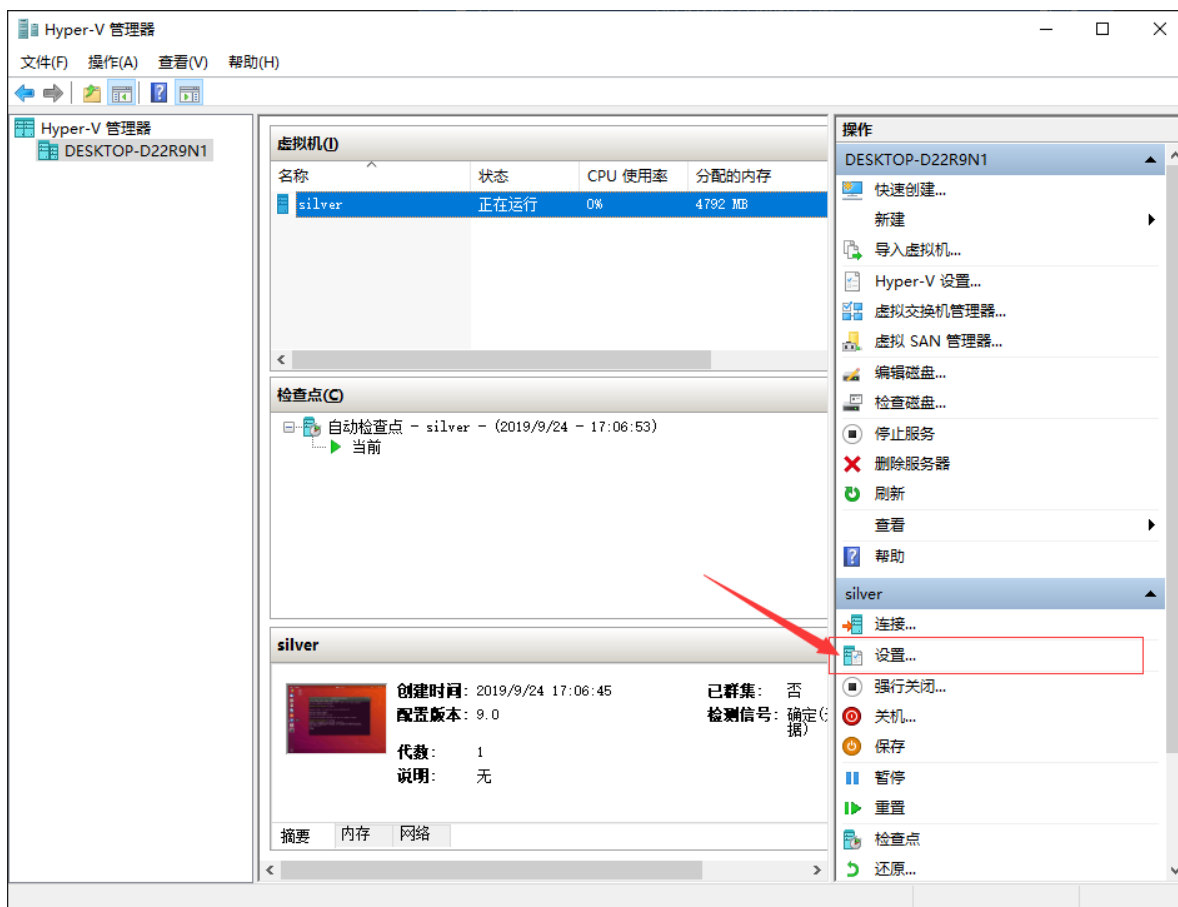




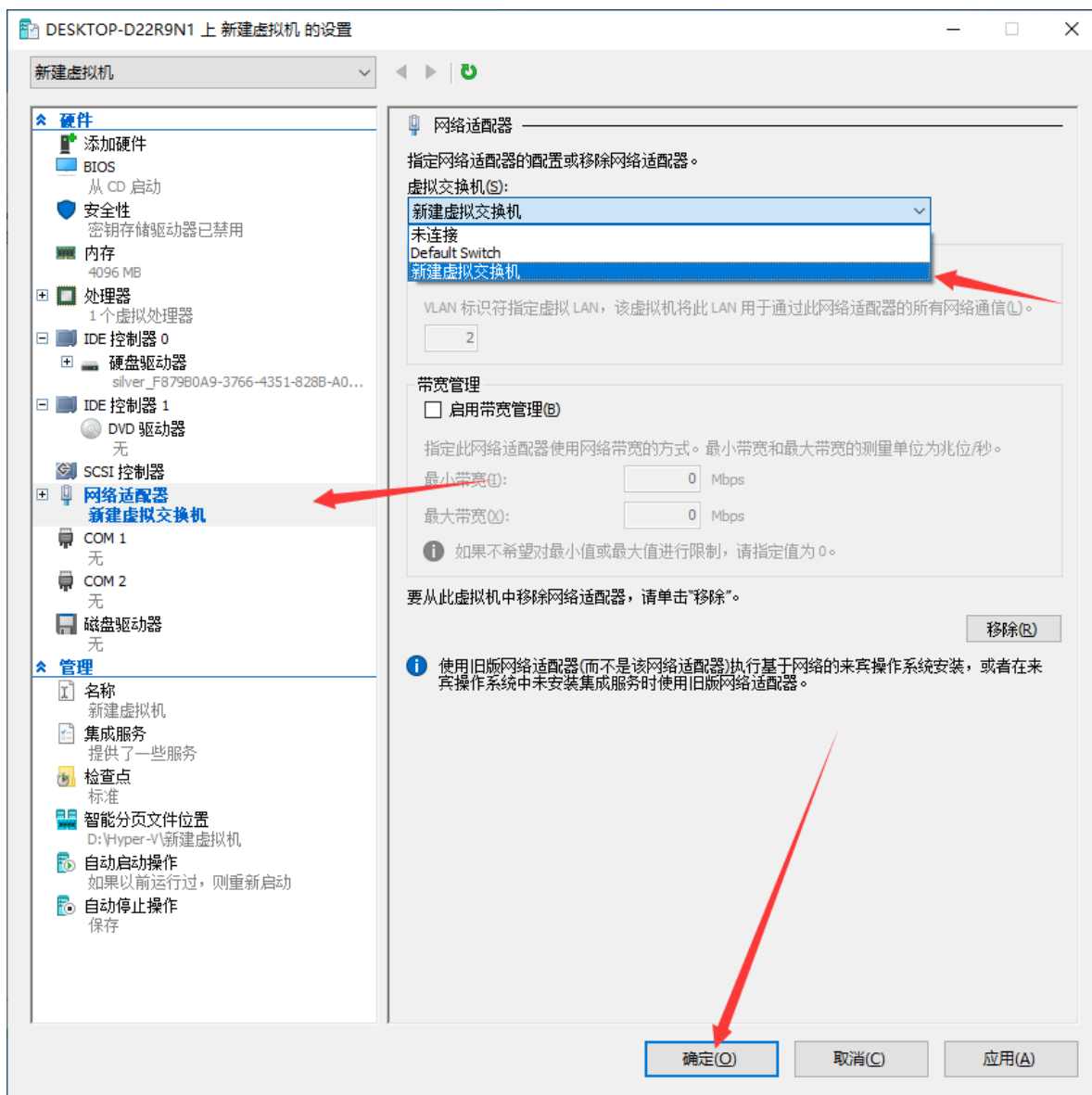


3. 等待应用更改

4. 在 Hyper-V 管理器中，选择虚拟机，点击右侧操作列表中的“设置” ，修改虚拟机配置。



5. 在虚拟机设置界面, 点击“网络适配器”, 选择虚拟交换机, 点击确定, 这样就配置好了虚拟机的网络连接。



这样就完成里虚拟机的网络配置了；

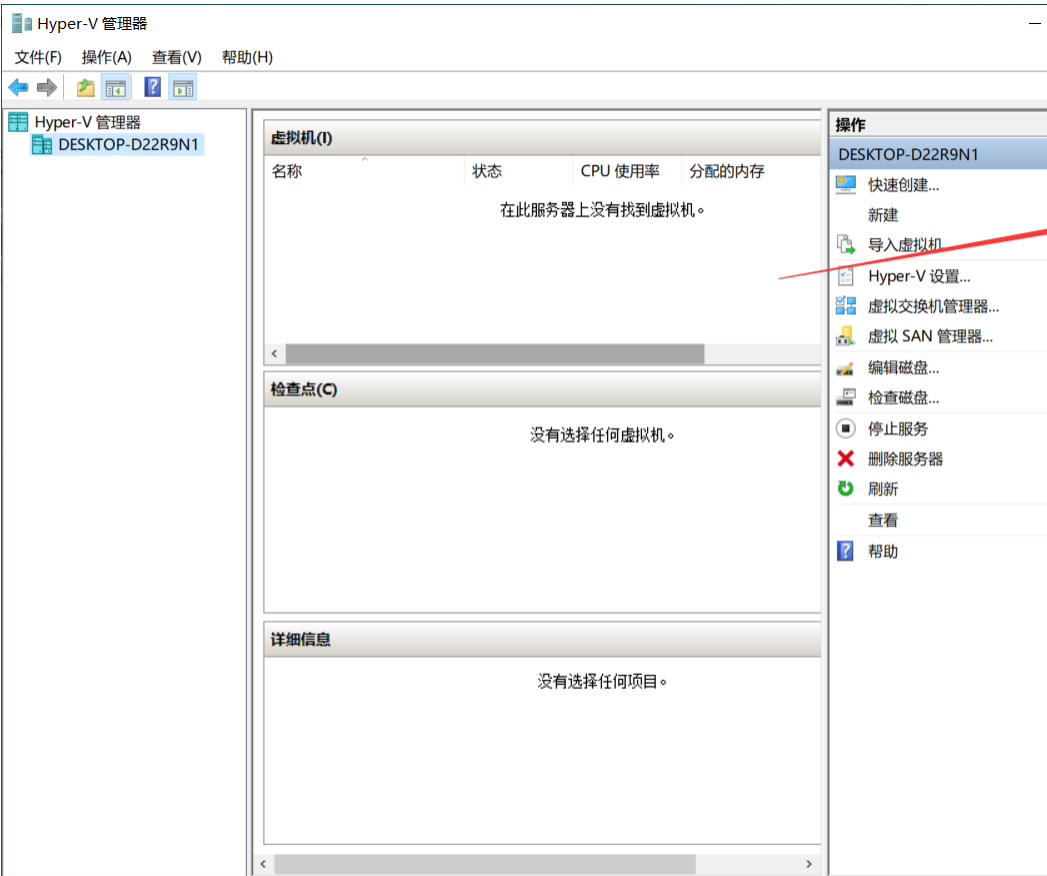
5.2.3 开始创建虚拟机了

1. 下载虚拟机需要的镜像

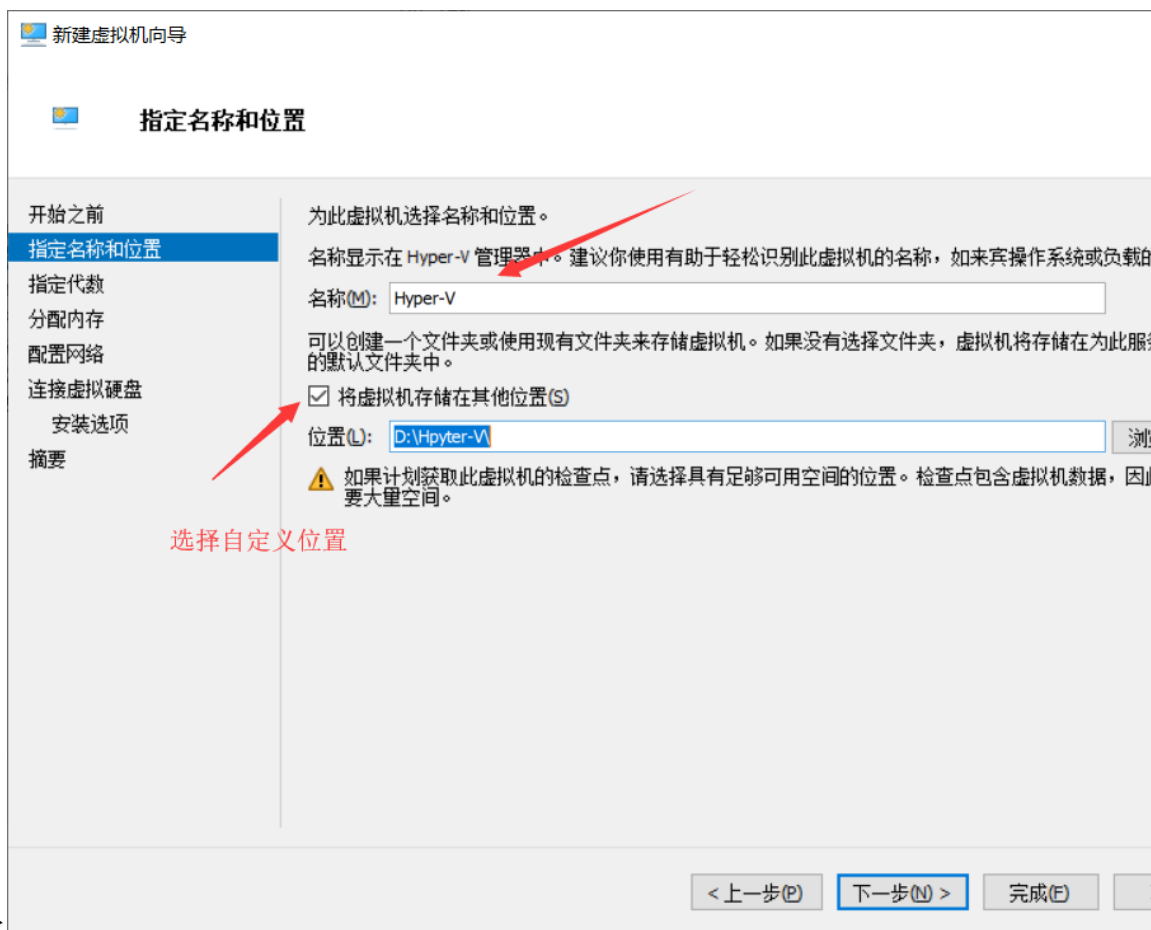
推荐 win10 镜像链接：<https://msdn.itellyou.cn/>

推荐 Linux 链接：<https://ubuntu.com/download/desktop/thank-you?country=CN&version=18.04.3&architecture=amd64#download>

将下载成功的镜像放在电脑上的一个文件夹中以备后用。



2. 开始创建 1) 点击新建虚拟机

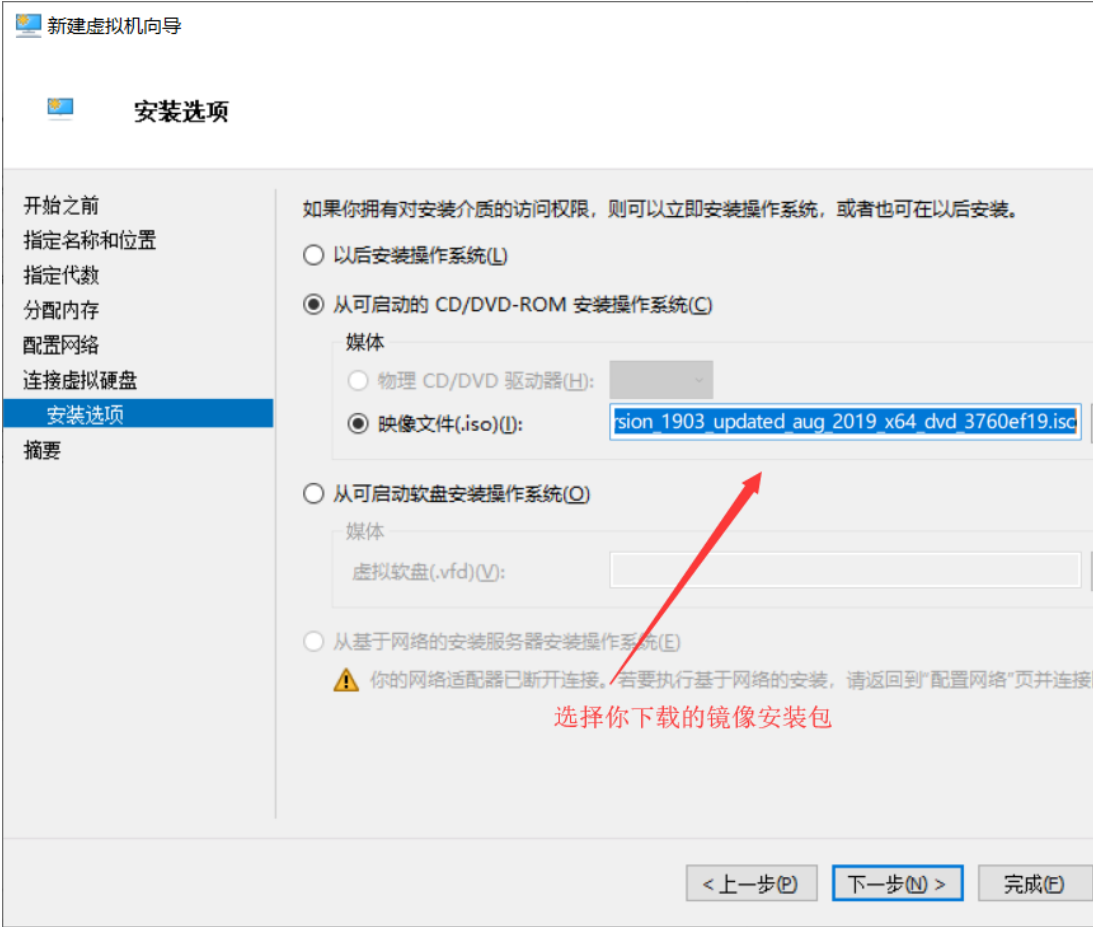


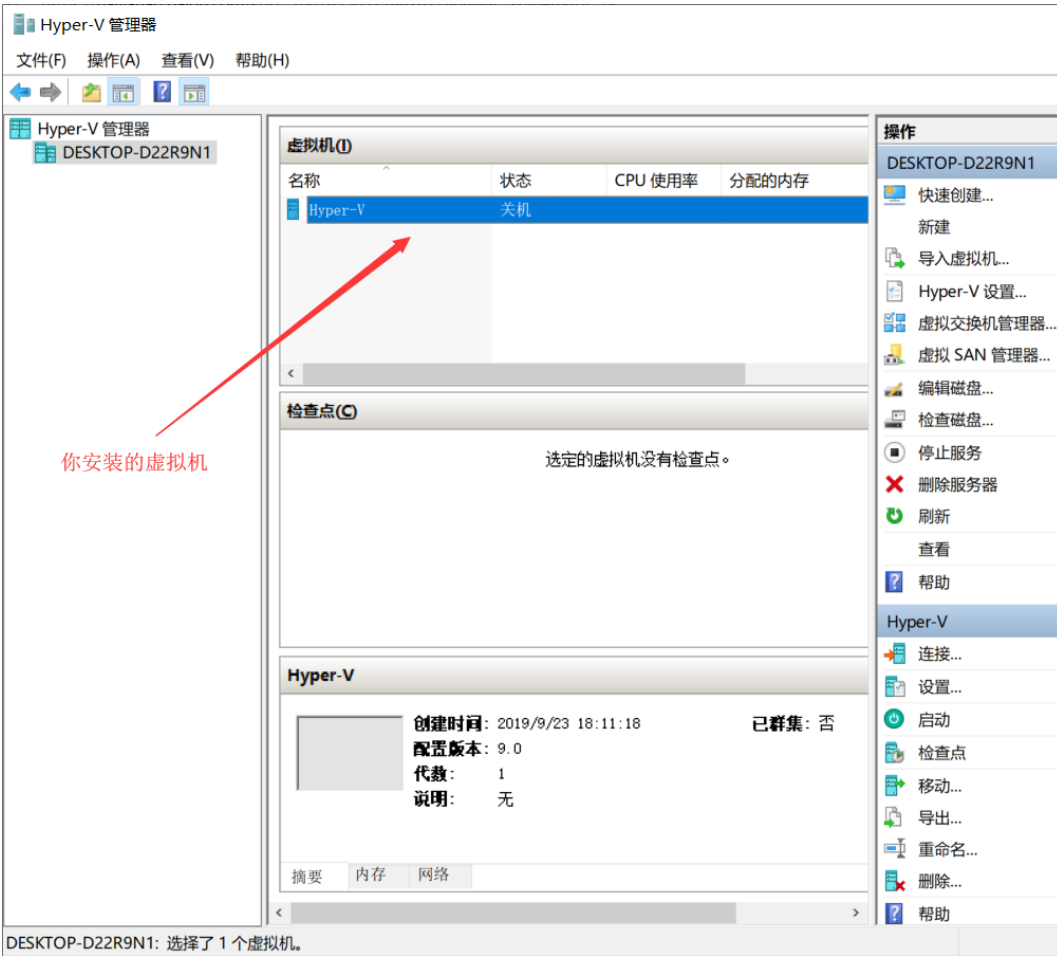
2) 设置你的虚拟机的名称

3) 指定代数，我选择的是第一代，继续选择下一步

4) 网络配置，选择我们刚刚设置的网络名称

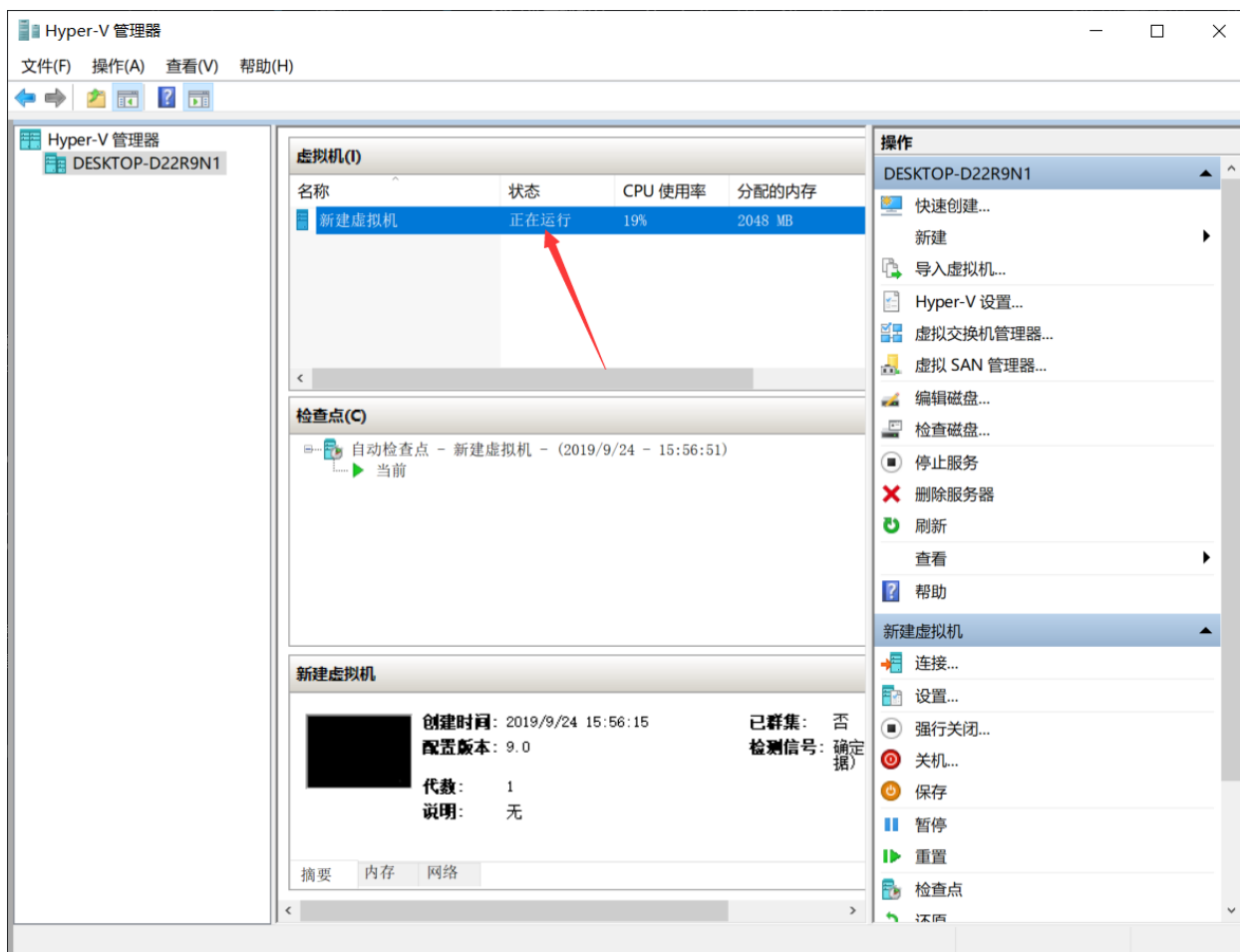
5)从我们的文件夹中选择镜像



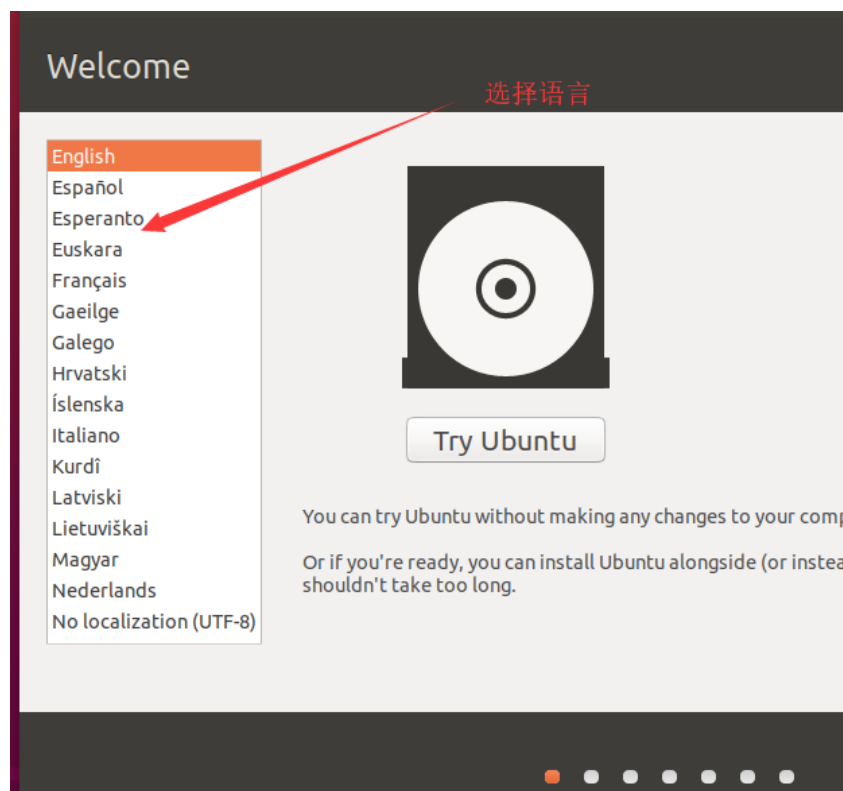


6)后续就是下一步,完成。示意图:

7) 这里显示已经创建好了, 然后右键选择启动该虚拟机 (有点小慢)。启动之后右键选择连接开始自动安装系统, 注意选择语言, 否则安装之后是英文, 如果对英文有点感冒的话, 这里就修改下, 要不然安装好之后有需要在设置里修改。



5.2.4 安排:



选择语言点击 `install ubuntu`, 之后就是下一步、下一步。

说明: 1. 因为是虚拟机所以可以清空虚拟 U 盘里的东西, 这样对你现有的电脑文件没有影响的 2. 如果重启的时候出现错误记得关闭虚拟机, 然后在开启。

5.2.5 参考:

1. <https://baijiahao.baidu.com/s?id=1607835632188281786&wfr=spider&for=pc>
2. <https://jingyan.baidu.com/article/a681b0de6109253b18434635.html>
3. <https://jingyan.baidu.com/article/e9fb46e129349d7520f76648.html>

5.3 进制转换题目解析

5.3.1 一、题目描述

我们可以用这样的方式来表示一个十进制数: 将每个阿拉伯数字乘以一个以该数字所处位置的 (值减 1) 为指数, 以 10 为底数的幂之和的形式。例如: 123 可表示十进制为 $1 \times 10^2 + 2 \times 10^1 + 3 \times 10^0$ 这样的形式。与之相似的, 对二进制数来说, 也可表示成每个二进制数乘以一个以该数字所处位置的 (值-1) 为指数, 以 2 为底数的幂之和的形式。一般说来, 任何一个正整数 R 或一个负整数 -R 都可以被选来作为一个数制系统的基数。如果是以 R 或 -R 为基数, 则需要用到的数码为 0, 1, ..., R-1。例如, 当 R=7 时, 所需用到的数码是

0, 1, 2, 3, 4, 5 和 6, 这与其是 R 或 $-R$ 无关。如果作为基数的数绝对值超过 10, 则为了表示这些数码, 通常使用英文字母来表示那些大于 9 的数码。例如对 16 进制数来说, 用 A 表示 10, 用 B 表示 11, 用 C 表示 12, 用 D 表示 13, 用 E 表示 14, 用 F 表示 15。在负进制数中是用 $-R$ 作为基数, 例如 -15 (十进制) 相当于 110001 (-2 进制), 并且它可以被表示为 2 的幂级数的和数:

$$110001 = 1(-2)^5 + 1(-2)^4 + 0(-2)^3 + 0(-2)^2 + 0(-2)^1 + 1(-2)^0$$

设计一个程序, 读入一个十进制数和一个负进制数的基数, 并将此十进制数转换为此负进制下的数: $-R \in \{-2, -3, -4, \dots, -20\}$ 。

输入

每个测试文件只包含一组测试数据, 每组输入两个整数, 第一个是十进制数 N ($-32768 \leq N \leq 32767$); 第二个是负进制数的基数 $-R$ 。

输出

对于每组输入数据, 输出此负进制数及其基数, 若此基数超过 10, 则参照 16 进制的方式处理。

输入样例 1

3000 -2

输出样例 1

30000=11011010101110000(base-2)

5.3.2 二、问题分析

涉及知识点: - 十进制数转化成二进制或者八进制或者 16 进制。- 短除法。- 数字的求模取余。

注意: 一个负数表示成二进制的方法可以先求它相反数的二进制数, 然后取反码, 最后取补码, 所得结果就是负数的二进制形式。不知道什么是原码、补码、反码的不要紧与解本题没有太大关系。

比较难为人的事情是, 题目中介绍了什么是二进制与十六进制, 但要我们求解一个负二进制到负二十进制。

简单介绍一下什么是短除法短除法是十进制转换成任意进制形式的计算方法。**十进制数转换成任意进制只要用短除法除对应的码数就行了**, 比如, 十进制数 1500 转换成二进制形式应该用短除法除 2, 如果要转成十六进制那应该短除 16, 转成二十进制应该短除 20。**十进制数 13 转换成二进制的计算过程:**

```

2 | 13  1
  | 6   0
  | 3   1
  | 1

```

所以 13 的二进制形式是 1101, 注意要从下往上写。短除法右边的值是余数, 13 除 2 等 6 余 1。

十进制数 -15 转换成负二进制的计算过程: 这个过程需要特别注意!

错误的运算:

$$\begin{array}{r} -2 \quad | \underline{-15} \quad -1 \\ \quad \quad 7 \end{array}$$

$$-1 = -15 - (-2 \times 7)$$

发现短除右边余数是-1, 显然错了, 二进制形式余数只能为 0 或 1, 同理十六进制余数只能为 0~15

正确的运算: 余数为负数时, 商应该 +1, 使得余数不为负数。

$$\begin{array}{r} -2 \quad | \underline{-15} \quad 1 \\ -2 \quad | \underline{8} \quad 0 \\ -2 \quad | \underline{-4} \quad 0 \\ -2 \quad | \underline{2} \quad 0 \\ -2 \quad | \underline{-1} \quad 1 \\ \quad \quad 1 \end{array}$$

$$1 = -15 - (-2 \times 8) = -15 + 16$$

$$0 = 8 - (-2 \times -4)$$

$$0 = -4 - (-2 \times 2)$$

$$0 = 2 - (-2 \times -1)$$

$$1 = -1 - (-2 \times 1)$$

所以-15 的负二进制为 110001。多问就会了。

所以我们把要做的事情分为以下几步: - 首先进行求模运算, 判断余数是否为负数 - 如果余数为负数, 则商+1, 再求余数, 把余数记录下来 - 如果余数不为负数, 正常求模运算, 把余数记录下来

为了方便转换, 定义一个字符数组 charList, 用于十六进制往后余数对应的字母 A、B~F。

Java 代码实现:

```
import java.util.ArrayList;
import java.util.Collections;
import java.util.List;
import java.util.Scanner;

public class Main {
    public static void main(String[] args) {

        Scanner sc = new Scanner(System.in);
        int num = sc.nextInt(); // 手动输入
        int base = sc.nextInt(); // 手动输入
        int num_copy = num;
```

(下页继续)

(续上页)

```

    int quotient = 0; // 商
    List<Character> list = new ArrayList<>(); // 动态数组来存, 因为不知道转换后有几位
    char[] charList = { '0', '1', '2', '3', '4', '5', '6', '7', '8', '9', 'A', 'B',
        ↪ 'C', 'D', 'E', 'F', 'G', 'H',
        'I', 'J' };
    while (num != 0) {
        int mod = num % base;
        if (mod < 0) {
            quotient = num / base + 1;
            mod = num - quotient * base;
            list.add(charList[mod]);
            num = quotient;
        } else {
            mod = num % base;
            num /= base;
            list.add(charList[mod]);
        }
    }
    Collections.reverse(list); // 数组倒序
    StringBuilder sb = new StringBuilder(256);
    for (int i = 0; i < list.size(); i++) {
        sb.append(list.get(i));
    }
    System.out.println(num_copy+"="+sb.toString() + "(base" + base + ")");
    sc.close();
}
}

```

C 代码实现:

```

#include <stdio.h>
#include <stdlib.h>
int main() {
    int num = 0;
    int base = 0;
    scanf("%d", &num); // 手动输入
    scanf("%d", &base); // 手动输入
    int num_copy = num;
    int quotient = 0; // 商

    char *list; // C 语言中的动态数组
    list = (char *)malloc(256 * sizeof(char));
    char charList[] = {'0', '1', '2', '3', '4', '5', '6', '7', '8', '9',

```

(下页继续)

(续上页)

```
        'A', 'B', 'C', 'D', 'E', 'F', 'G', 'H', 'I', 'J'};

    int i = 0; //计数器
    while (num != 0) {
        int mod = num % base;
        if (mod < 0) {
            quotient = num / base + 1;
            mod = num - quotient * base;
            list[i] = charList[mod];
            num = quotient;
            i++;
        } else {
            mod = num % base;
            num /= base;
            list[i] = charList[mod];
            i++;
        }
    }

    int k = 0;
    printf("%d=", num_copy);
    for (int i = 256; i >= 0; i--) {
        if (list[i] == NULL) {
            continue;
        }
        printf("%c", list[i]);
    }
    printf("(base%d)", base);
    return 0;
}
```

5.4 几种常见的博客搭建资源汇总

本文首发于: [博客搭建资源汇总](#) 作者: [LvLv](#) 略有修改

5.4.1 前言

本文给出了几种常见的个人网站（博客）搭建方式的教程资源。官方文档为主，B 站视频教程为辅。比较适合喜欢折腾的同学。不喜欢折腾的可以直接使用 CSDN 博客、博客园、语雀等提供的服务。别忘了博客的初心是记录。

常见的个人博客、网站、知识库搭建方式如下：



动态管理系统需要使用服务器部署，因为这些系统的数据大都存放在数据库。现今大部分的云服务商（阿里云、AWS、腾讯云等等）大都提供了一个月甚至一年的服务器试用期。

依赖静态站点生成器生成的网站不仅可以部署在服务器上，还可以利用一些免费的站点托管服务进行托管，比如GitHub、Coding、Gitlab等提供的 Pages 服务和Read the Docs的文档托管服务。深入折腾的同学还会接触到一些CI/CD工具链、SEO技巧等等。

对于 Web 开发者，你可能会接触到一些 BASS 服务提供商，比如LeanCloud、Bmob、firebase。

5.4.2 热门方式

1.WordPress 搭建个人博客

首先根据维基百科来介绍一下 WordPress：

WordPress 是一个以PHP和MySQL为平台的自由开源的博客软件和内容管理系统。WordPress 具有插件架构和模板系统。截至 2018 年 4 月，排名前 1000 万的网站中超过 30.6% 使用 WordPress。WordPress 是最受欢迎的网站内容管理系统。WordPress 是当前因特网上最流行的博客系统。WordPress 在最著名的网络发布阶段中脱颖而出。如今，它被使用在超过 7000 万个站点上。

步骤：

- 1. 注册域名
- 1. 创建 VPS（虚拟专用服务器，即需要购买服务器）
- 1. 解析域名
- 1. 绑定域名

- 1. 上传配置文件
- 1. 安装使用

具体详细操作详见如下文档和视频:

[wordpress 搭建博客](#)

[教你用 WordPress 搭建一个简单的个人博客](#)

[免费搭建自己的 wordpress 博客 < 视频教程 >](#)

[使用 WordPress 搭建博客网站](#)

2.Typecho 搭建个人博客

首先介绍一下 Typecho:

Typecho 是一个基于PHP的简洁的开源博客程序。它使用多种数据库 (MySQL、PostgreSQL、SQLite、MariaDB) 储存数据, 在GNU GPLv2许可证下发行

- 搭建步骤:
 - 1. 注册域名
 - 1. 购买 vps
 - 1. 解析域名
 - 1. 环境安装
 - 1. 安装 typecho

具体详细操作流程可参照以下文章和视频:

[typecho 文档](#)

[typecho 搭建自己博客网站视频](#)

[使用 typecho 搭建个人博客视频](#)

3. 使用 Hugo 搭建个人博客

详细介绍请参照 Hugo 文档:

[Hugo 中文文档](#)

[Hugo 博客搭建视频教程](#)

[使用 Hugo 搭建个人博客](#)

4.Hexo+GitHub 搭建博客

Hexo 官方文档

Hexo 视频教程

ShanSan 的好友 TRHX 的手把手小白教程

5.Jekyll 搭建个人博客

Jekyll 文档

Jekyll 搭建个人博客视频教程

6.GitBook

GitBook 文档

GitBook 视频

7.Ghost 搭建博客

Ghost 官方文档

Ghost 搭建个人博客视频教程

8.VuePress 搭建个人博客

VuePress 中文文档

VuePress 搭建个人博客

9.docsify 搭建个人文档

docsify 官方文档

5.4.3 创新基地已知个人博客一览

- 国禾回星
- ShanSan
- Zh1
- LvLv
- 兴杏梅

- [nornoya](#)
- [Suzee_](#)
- [pengliz](#)
- [Y-lin9](#)
- [yuchendong](#)

博客、个人网站、知识库需要时常维护，收益不是立马就能看到的。。有些同学弄可能只是因为一时的新鲜感，说实话，这样起不到任何效果，啥也学不到。

顺便再提一句，没加 **GitHub** 组织的同学快留下你的邮箱：<https://github.com/seven-innovation-base/SphinxDOC/issues/7>

CHAPTER 6

干货推荐

注解：部分文章需要科学上网

6.1 程序员能力矩阵

<http://sijinjoseph.com/programmer-competency-matrix/>

6.2 科学上网

- <https://github.com/bannedbook/fanqiang/wiki>
- <https://fanqiang.software-download.name/>
- <https://shan333.cn/2018/10/07/Proxy/>

6.3 C/C++ 相关

- Standard C++
- cppreference

6.4 跟我一起写 Makefile

- <https://blog.csdn.net/haoel/article/details/2886>
- <https://blog.csdn.net/haoel/article/details/2887>
- <https://blog.csdn.net/haoel/article/details/2888>
- <https://blog.csdn.net/haoel/article/details/2889>
- <https://blog.csdn.net/haoel/article/details/2890>
- <https://blog.csdn.net/haoel/article/details/2891>
- <https://blog.csdn.net/haoel/article/details/2892>
- <https://blog.csdn.net/haoel/article/details/2893>
- <https://blog.csdn.net/haoel/article/details/2894>
- <https://blog.csdn.net/haoel/article/details/2895>
- <https://blog.csdn.net/haoel/article/details/2896>
- <https://blog.csdn.net/haoel/article/details/2897>
- <https://blog.csdn.net/haoel/article/details/2898>
- <https://blog.csdn.net/haoel/article/details/2899>

6.5 RSA 算法原理

- <https://blog.csdn.net/u014044812/article/details/80723009>
- <https://blog.csdn.net/u014044812/article/details/80782448>
- <https://blog.csdn.net/u014044812/article/details/80866759>

6.6 UI/UX

- 7 steps to become a UI/UX designer

6.7 Git 相关

- 团队中的 Git 实践
- A successful Git branching model
- 使用 git rebase 避免無謂的 merge
- Git: 教你如何在 Commit 时有话可说

6.8 More

- 关于模运算的思考

7.1 入门必知

- 算法：解决某个问题的计算方法、步骤。《如何通俗易懂的解释什么是算法》
- 關於字符編碼，你所需要知道的
- CLI(Command Line Interface)：CMD、Windows PowerShell、Bash、...

注解：通常认为，命令行界面（CLI）没有图形用户界面（GUI）那么方便用户操作。因为，命令行界面的软件通常需要用户记忆操作的命令，但是，由于其本身的特点，命令行界面要较图形用户界面节约计算机系统的资源。在熟记命令的前提下，使用命令行界面往往要较使用图形用户界面的操作速度要快。所以，在现在的图形用户界面的操作系统中，通常都保留着可选的命令行界面。虽然现在许多计算机系统都提供了图形化的操作方式，但是却都没有因而停止提供文字模式的命令行操作方式，相反的，许多系统反而更加强这部分的功能，例如 Windows 就不只加强了操作命令的功能和数量，也一直在改善 Shell Programming 的方式。而之所以要加强、改善，自然是因为不够好；操作系统的图形化操作方式对单一客户端计算机的操作，已经相当方便，但如果是一群客户端计算机，或者是 24 小时运作的服务器计算机及路由器、交换器等电信设备，图形化操作方式有时会力有未逮，所以需要不断增强命令行界面的脚本语言和宏语言来提供丰富的控制与自动化的系统管理能力，例如 Linux 系统的 Bash 或是 Windows 系统的 Windows PowerShell。- 引自维基百科

注解：RFC，请求意见稿（英语：Request for Comments，缩写：RFC）是由互联网工程任务组（IETF）发布的一系列备忘录。文件收集了有关互联网相关信息，以及 UNIX 和互联网社群的软件文件，以编号排定。当前 RFC 文件是由互联网协会（ISOC）赞助发行。RFC 始于 1969 年，由当时就读加州大学洛杉矶分校（UCLA）

的斯蒂芬·克罗克 (Stephen D. Crocker) 用来记录有关 ARPANET 开发的非正式文档, 他是第一份 RFC 文档的撰写者。最终演变为用来记录互联网规范、协议、过程等的标准文件。基本的互联网通信协议都有在 RFC 文件内详细说明。RFC 文件还额外加入许多的论题在标准内, 例如对于互联网新开发的协议及发展中所有的记录。

7.2 信息安全相关

- CTF 夺旗赛

注解: CTF (Capture The Flag, 夺旗赛) 是一种流行于网络安全技术人员之间的一种信息安全技术竞赛。其前身是传统黑客之间网络技术比拼的游戏, 以代替之前黑客们通过互相发起真实攻击进行技术比拼的方式。起源于 1996 年第四届 DEFCON。现在已成为全球范围网络安全圈流行的竞赛形式。

通常 CTF 分为三种赛制: 解题赛 (Jeopardy)、攻防赛 (Attack-Defence) 和混合赛。

竞赛模式基本分为解题模式、攻防模式和混合模式。大多题目的内容基本包括 web 安全, 密码学、逆向、二进制安全编程类题目等国内外有很多 CTF 比赛。

解题赛是线上赛通常采取的赛制, 题目通常分为多个类型, 如 Web, Forensic (取证), Crypto (密码学), Binary (二进制) 等。团队或个人可以通过解题获得一串具有一定格式的字符串, 也就是 flag。将 flag 提交到竞赛平台可以获得积分。题目的难度越大, 分值就越高。当比赛结束后, 得分最高者胜出。

攻防赛是另一种有趣的赛制, 常见于线下决赛。攻防赛中, 每个队伍都会被分配一台主机或虚拟机, 称为 gamebox, 队员可以通过网络连接到 gamebox。而所有队伍的 gamebox 通过内网连接在一起。每个队伍的 gamebox 上都运行着多个相同的服务。参赛队伍需要挖掘服务的漏洞, 然后攻击其他队伍的服务来获取 flag, 并提交给计分服务器来获取积分。与此同时, 参赛队伍还需要修补自身服务中的漏洞来防止丢分。攻防赛不仅考验了参赛选手的技术水平, 还考验了参赛者的体力, 因为通常参赛者需要连续混合赛可能采取解题赛和攻防赛的混合模式, 也可能是其他形式。

- CTF Wiki

<https://ctf-wiki.github.io/ctf-wiki/>

7.3 大数据技术相关

7.4 软件开发相关

- 模块化程序设计、分而治之

模块化程序的编写不是开始就逐条录入计算机语句和指令, 而是首先用主程序、子程序、子过程等框架把软件的主要结构和流程描述出来, 并定义和调试好各个框架之间的输入、输出链接关系。逐步求精的结果是得

到一系列以功能块为单位的算法描述。以功能块为单位进行程序设计，实现其求解算法的方法称为模块化。模块化的目的是为了降低程序复杂度，使程序设计、调试和维护等操作简单化。

- 高内聚、低耦合

高内聚：模块内各元素关联性强，紧密程度高。

低耦合：让每个模块，尽可能的独立完成某个特定的子功能。

高内聚，低耦合的好处体现在系统持续发展的过程中，高内聚，低耦合的系统具有更好的重用性，维护性，扩展性，可以更高效的完成系统的维护开发，持续的支持业务的发展，而不会成为业务发展的障碍。

图解高内聚低耦合

7.5 前端相关

- 浏览器兼容性查询

8.1 Markdown 语法渲染测试

CommonMark风格, parser->commonmarkpy、recommonmark

8.1.1 什么是 Markdown

Markdown 是一种方便记忆、书写的纯文本标记语言，用户可以使用这些标记符号以最小的输入代价生成极富表现力的文档：譬如您正在阅读的这份文档。它使用简单的符号标记不同的标题，分割不同的段落，**粗体**或者 *斜体*某些文字，更棒的是，它还可以

1. 制作列表

- I
 - II
- III
 - IV
 - V
 - Vi

2. 高亮一段代码

```
@requires_authorization
class SomeClass:
    pass

if __name__ == '__main__':
    # A comment
    print 'hello world'
```

4、快速插入图片、链接

<https://github.com/seven-innovation-base/SphinxDOC>

《和平宣言》

巍巍金陵，滔滔大江，钟山花雨，千秋芬芳。
一九三七，祸从天降，一二一三，古城沦丧。
侵华日寇，掳掠烧杀，尸横遍野，血染长江。
三十余万，生灵涂炭，炼狱六周，哀哉国殇。
举世震惊，九州同悼，雪松纪年，寒梅怒放。
亘古浩劫，文明罹难，百年悲叹，警钟鸣响。
积贫积弱，山河蒙羞，内忧外患，国破家亡。
民族觉醒，独立解放，改革振兴，国运日昌。
前事不忘，后事之师，殷忧启圣，多难兴邦。
八十二载，青史昭彰，生生不息，山高水长。
二零一九，国家公祭，中外人士，齐聚广场。
白花致哀，庄严肃穆，丹忱抒写，和平诗章。
大道之行，天下为公，大德曰生，和气致祥。
和平发展，时代主题，民族复兴，世代梦想。
龙盘虎踞，彝训鼎铭，继往开来，永志不忘。

5、更多精彩

观看本页 [Markdown 源格式](#)

你只需使用键盘专注于书写文本内容，就可以生成印刷级的排版格式，省却在键盘和工具栏之间来回切换，调整内容和格式的麻烦。**Markdown** 在流畅的书写和印刷级的阅读体验之间找到了平衡。目前它已经成为世界上最大的技术分享网站 [GitHub](#) 和技术问答网站 [StackOverflow](#) 的御用书写格式。

8.2 rst 渲染测试

reStructuredText 是 Sphinx 使用的默认纯文本标记语言。[RST 初级读本](#)

8.2.1 显式标记

[了解更多](#)

警告： 警告

危险： 危险

错误： 错误

提示： 提示

重要： 重要

注解： 注释

小技巧： 技巧

警告: 警告

8.2.2 原始标记格式

8.2.3 数学公式 LaTeX 支持

$$(a + b)^2 = a^2 + 2ab + b^2$$

$$(\alpha \times \beta)^2 = a^2 - 2ab + b^2$$

LaTeX 支持情况

[查看本页 reStructuredText 源格式](#)

本文档使用 Sphinx + ReadtheDocs + GitHub 搭建，所使用的文本标记语言为 reStructuredText + Markdown

如果你想参与文档的编辑或者发现文档中有错误，麻烦提交个 [issue](#) 给我们，或者 fork 本项目，向我们提交 pull request，具体如何操作请阅读下本项目的 [README](#)

9.1 参考资源

- [Sphinx 官方中文文档](#)
- [ReadtheDocs](#)
- [使用 pandoc markdown 在线转 reStructuredText](#)
- [使用 ReadtheDocs 托管文档](#)