中国科学院大学
University of Chinese Academy of Sciences

计算机科学导论实验

个人作品-1
制作静态网页

zxu@ict.ac.cn
zhangjialin@ict.ac.cn

目标

- 使用创造性的表达制作**动态**的网页。
- 完成本实验需要同学们展示**自学**的能力。
- 每位同学需要完成并**向全班展示**：
  - 一个**动态**的网页，包含HTML，CSS，JavaScript代码
  - 该网页展示了你的**创造力**

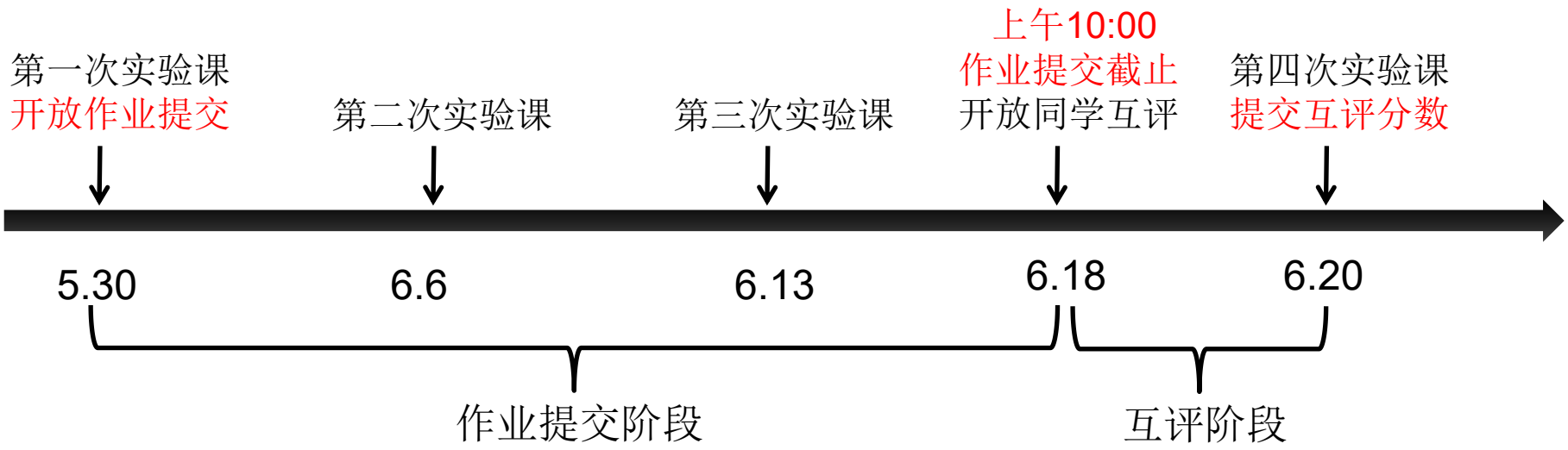
个人作品实验安排

周次	日期	实验课	课程内容	作业提交内容	截止时间
14	5月30日	制作静态网页	HTML结构 HTML语法	作品名称、包含index.html的zip文件	6月18日 10:00前
15	6月6日	制作网页版计算器	修改计算器的结果 读取操作数 执行选中的运算		
16	6月13日	制作网页版时钟	添加静态时钟 让时钟动起来 Debug方法 对比JS和Go		
17	6月20日	网页打分	小组讨论与互评	评分结果	

本课程将会以“王小二”编写一个网页博客介绍他发明了网页版时钟和计算器的伟大事迹为例，讲述相关的编程知识。

评分标准

- 6月18日 10:00 前可反复提交并预览
- 教师评分+同学互评
  - 同学互评开放时间：6月18日 17:00 - 6月20日 13:30
  - 评分维度：创造性与完成度



评分标准

- 创造性
  - 0分：无创意，完全依赖模板或已有内容，无任何创新或个性化表达。
  - 1分：创意欠缺，偶有微小改动，但整体仍以模仿为主，尝试加入少量新元素，但效果不明显或逻辑牵强。
  - 2分：具备基础创新性，结合现有方法或内容，有一定重组或改进。
  - 3分：较有创意，内容或设计具有独特性和启发性。
  - 4分：很有创意，且体现了精心的设计与实现。

评分标准

- 完成度
  - 0分：未完成，无法展示核心功能，或存在抄袭，使用他人材料无致谢。
  - 1分：完成度低，有严重缺陷，仅能展示很少部分功能，使用他人材料或代码比例过大。
  - 2分：基本完成，有bug，能展示主要功能，多处使用他人资源但致谢规范。
  - 3分：完成，仅有少量bug，几乎不影响展示功能，使用少量外部资源且致谢规范。
  - 4分：完成度高，没有bug，功能展示完整，几乎完全原创，致谢规范。

作品提交

- 时间：2025 年 6 月 18 日 10:00 前
- 提交内容：作品名称 + zip压缩包
  - 在https://part.cs101.cs.ac.cn/submission的作品提交界面提交包含index.html的zip压缩包及作品名称

参考

- 在线教程
  - https://www.w3schools.com/
  - https://www.freecodecamp.org/chinese/learn
- 实验参考示例与部分往年作品

个人作品 参考示例 作业详情

音乐类

猫猫指挥家 小小演奏家 韵律

算术类

四则运算_1 四则运算_2 四则运算_3 四则运算_4 二进制乘法
显示pi_1 显示pi_2 显示pi_3 显示pi_4 IEEE-754 转换器

棋盘类

棋盘_1 棋盘_2 棋盘_3 棋盘_4 随机游走_1 随机游走_2 随机游走_3 随机游走_4 像素点阵图

2023-2024学年春季学期部分作品展示

Fantasy On Ice

自制谱面音游/钢琴模拟器

花序钟

排序可视化

走！去海淀

通过傅里叶变换绘制果壳和ಠ◡ಠ?

你的编程可以很抽象！

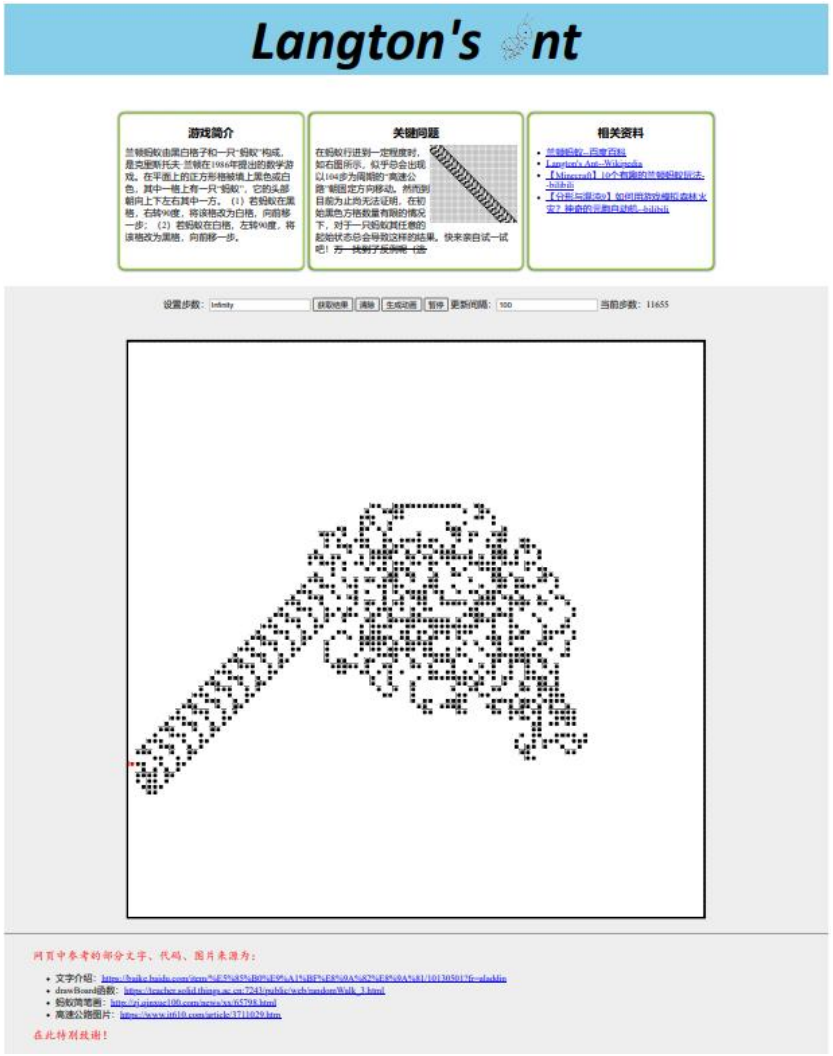
在康威生命游戏中显示计科导

新汉诺塔问题最优解可视化

注意事项

- 本实验不允许使用Vue，React等外部框架，**外部代码占比<10%**
- 致谢以文字的形式在页面底部显示（而不是注释）
  - 如果引用外部图片等资源，必须明确指明图片等资源链接
- 标注创造性：在<head>部分以注释的形式说明创造性
- 本次实验重在进行“**创造性表达**”
  - 评分维度中不包含“实现复杂度”
  - 并非实现的越复杂越“好”
  - 并非实现的难度越高越“好”
  - 并非代码量越多越“好”
  - 并非复现一个复杂的小游戏就是“有创造性的作品”
  - 建议结合计科导课程/个人专业/社会问题等进行选题

例子–结合你的专业



Langton's_Ant.html

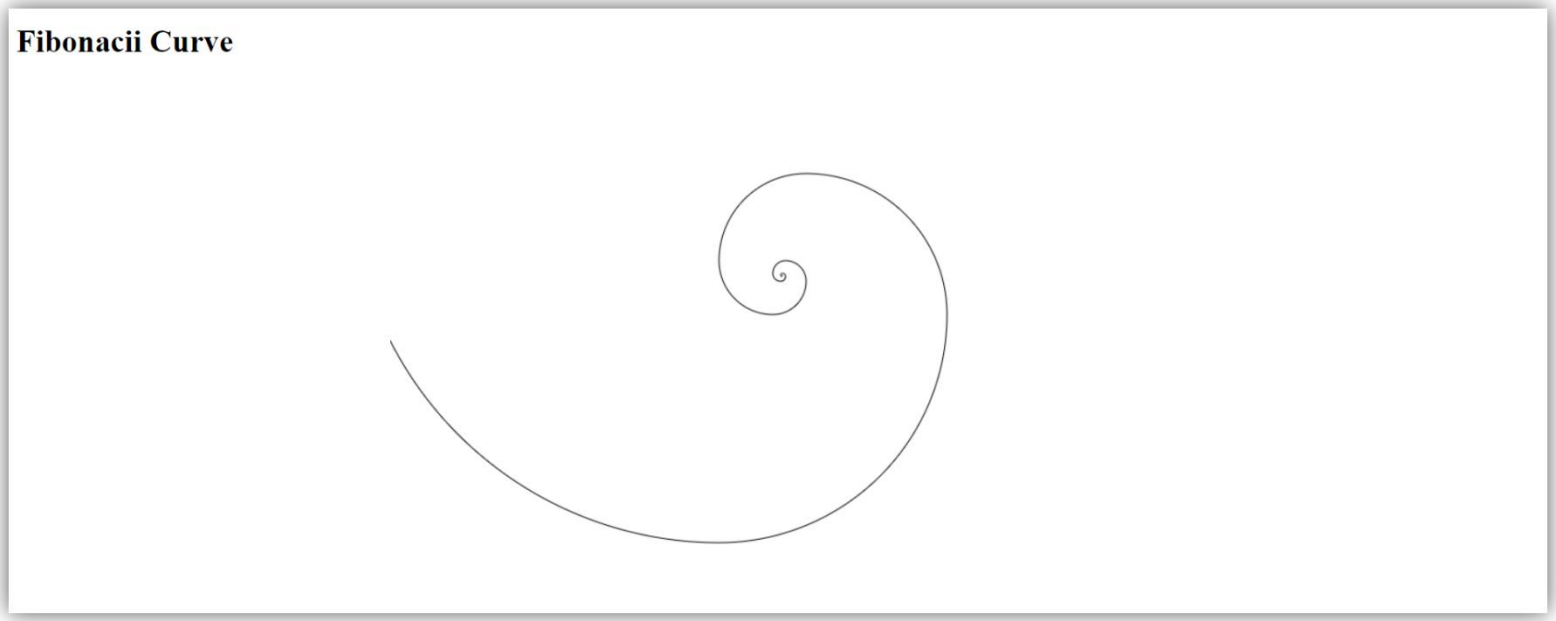
Langton’s Ant（兰顿蚂蚁）是一个数学游戏

例子–结合你的专业

通过傅里叶变换绘制果壳和ʚ◡ɞ?

例子-编程习惯好

- 此网页展示了斐波那契亚曲线
- 遵循了良好的编程习惯



Fibonacci.html

例子–编程习惯好

```
// code executed directly:
var len = 14; // the number of Fibonicaii numbers to form the Curve (Global variable)
var fibonaciiArray = new Array(len); // Fibonicaii numbers to form the Curve
calculate_fibnacii();
drawFibonacciArc();
// function definition:
function calculate_fibnacii(){
    fibonaciiArray[0] = 1; fibonaciiArray[1] = 1;
    for(var i=2;i<len;i++) fibonaciiArray[i]=fibonaciiArray[i-1]+fibonaciiArray[i-2];
}
function drawFibonacciArc(){
    // show the whole curve by draw many 1/4 circles
    var canvas = document.getElementById("myCanvas"); // get Canvas
    var ctx = canvas.getContext("2d");
    var x0 = 400, y0 = 200;
    var x = x0, y = y0, startDegree = 0, endDegree = 90;

    for (var i = 0; i < len; i++) {
        // Draw 1/4 Circle whose radius is fibonaciiNumber[i]
        var radius = fibonaciiArray[i];
        ctx.beginPath();
        ctx.strokeStyle = "black";
        ctx.arc(x,y,radius,(startDegree/180)*Math.PI,(endDegree/180)*Math.PI);
        ctx.stroke();
        // update the position of the center of next 1/4 circle
        var direction = startDegree/90;
        if(i<len-1){
            var inc_x = (direction%2-2*Math.floor(direction/3))*(fibonaciiArray[i+1]-fibonaciiArray[i]);
            var inc_y = -(1-direction+2*Math.floor(direction/3))*(fibonaciiArray[i+1]-
fibonaciiArray[i]);
            x += inc_x; y += inc_y;
        }
        // update the range of degrees of next 1/4 circle
        startDegree = (startDegree+90)%360;
        endDegree = (endDegree+90)%360;
    };
}
```

- 把常数的定义放在前面
- 使用描述性的变量名
- 避免重复的代码
- 使用注释
- 避免“魔数”

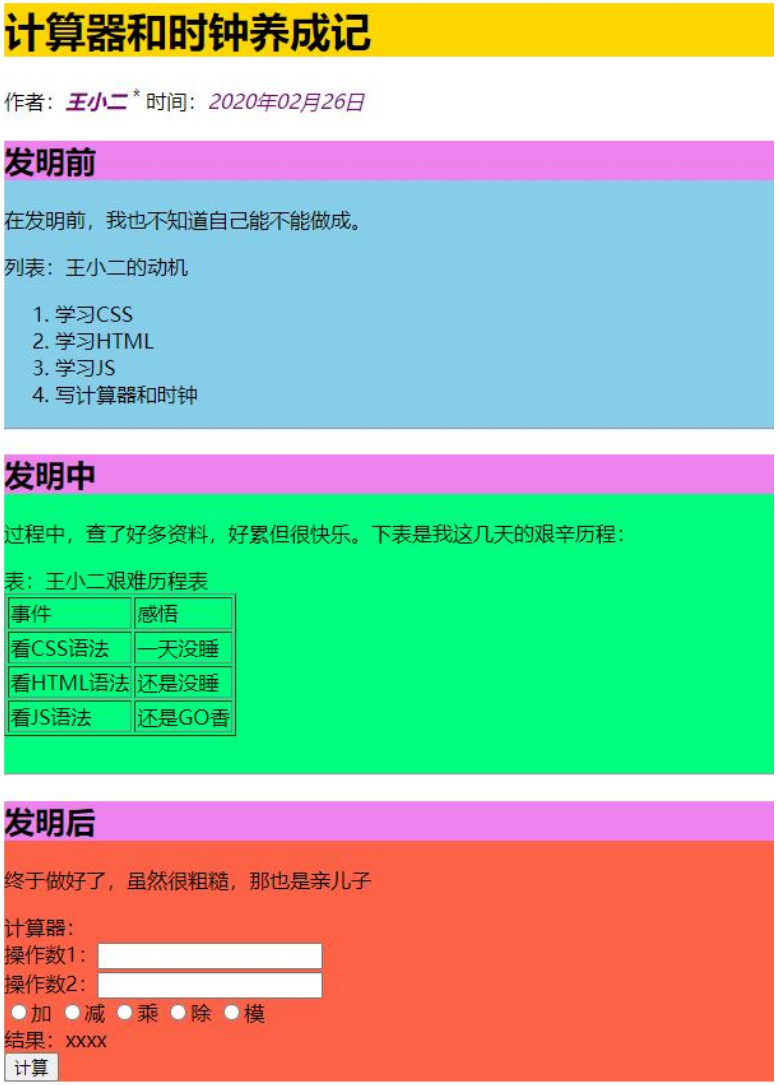
例子-Tips

```
// code executed directly:
var len = 14; // the number of Fibonicaii numbers to form the Curve (Global variable)
var fibonaciiArray = new Array(len); // Fibonicaii numbers to form the Curve
calculate_fibnacii();
drawFibonacciArc();
// function definition:
function calculate_fibnacii(){
    fibonaciiArray[0] = 1; fibonaciiArray[1] = 1;
    for(var i=2;i<len;i++) fibonaciiArray[i]=fibonaciiArray[i-1]+fibonaciiArray[i-2];
}
function drawFibonacciArc(){
    // show the whole curve by draw many 1/4 circles
1.  var canvas = document.getElementById("myCanvas"); // get Canvas
2.  var ctx = canvas.getContext("2d");
    var x0 = 400, y0 = 200;
    var x = x0, y = y0, startDegree = 0, endDegree = 90;

    for (var i = 0; i < len; i++) {
        // Draw 1/4 Circle whose radius is fibonaciiNumber[i]
        var radius = fibonaciiArray[i];
        ctx.beginPath();
3.      ctx.strokeStyle = "black";
        ctx.arc(x,y,radius,(startDegree/180)*Math.PI,(endDegree/180)*Math.PI);
4.      ctx.stroke();
        // update the position of the center of next 1/4 circle
        var direction = startDegree/90;
        if(i<len-1){
            var inc_x = (direction%2-2*Math.floor(direction/3))*(fibonaciiArray[i+1]-fibonaciiArray[i]);
            var inc_y = -(1-direction+2*Math.floor(direction/3))*(fibonaciiArray[i+1]-fibonaciiArray[i]);
            x += inc_x; y += inc_y;
        }
        // update the range of degrees of next 1/4 circle
        startDegree = (startDegree+90)%360;
        endDegree = (endDegree+90)%360;
    };
}
```

- Canvas 的用法:
  1. canvas = document.getElementById("…")
  2. ctx = canvas.getContext("2d")
  3. ctx.XXX()
  4. ctx.stroke()

本节课制作：



计算器和时钟养成记

作者：王小二* 时间：2020年02月26日

发明前

在发明前，我也不知道自己能不能做成。

列表：王小二的动机

1. 学习CSS
2. 学习HTML
3. 学习JS
4. 写计算器和时钟

发明中

过程中，查了好多资料，好累但很快乐。下表是我这几天的艰辛历程：

表：王小二艰难历程表

事件	感悟
看CSS语法	一天没睡
看HTML语法	还是没睡
看JS语法	还是GO香

发明后

终于做好了，虽然很粗糙，那也是亲儿子

计算器：
操作数1：
操作数2：
加 减 乘 除 模
结果：xxxx
计算

index.html

第一周 提纲

- HTML 结构
- HTML 语法

HTML：超文本标记语言
英文全称：HyperText Markup Language
这种语言用来描述网页上有什么

HTML结构

```
<html>
        <head>
        一些代码
        </head>
        <body>
        一些代码
        </body>
</html>
```

所有 HTML 代码都需要遵循这一结构

注意 “/“ 的存在!

HTML 结构–例子

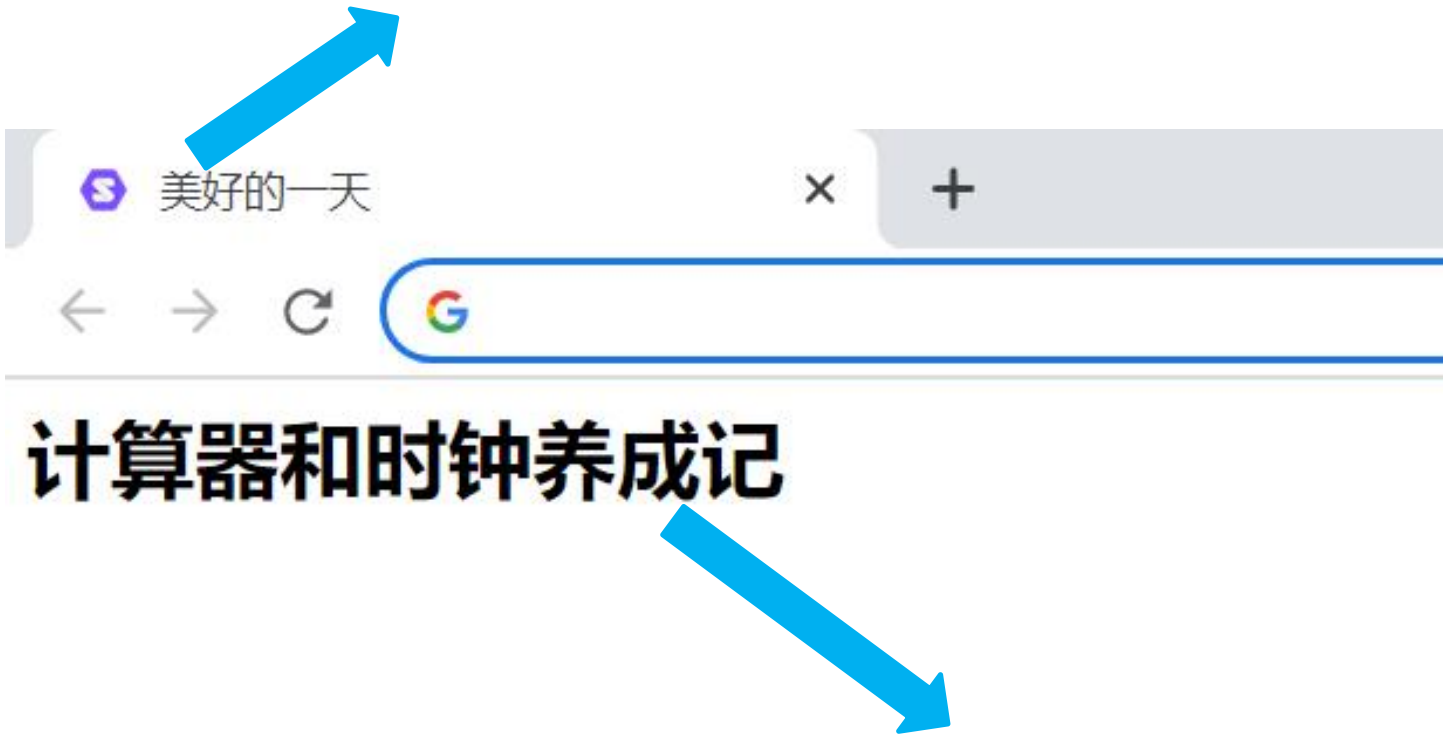
新建 index.html 文档，并写入下面的代码：

[代码意义稍后讲述]

```
<html>
    <head>
        <meta charset="UTF-8" >
        <title>美好的一天</title>
    </head>
    <body>
        <h1>计算器和时钟养成记</h1>
    </body>
</html>
```

使用浏览器(建议chrome)打开上述文档

<title>美好的一天</title>



<h1>计算器和时钟养成记</h1>

HTML 结构–例子

- 所有的html源代码的文件名均以“.html”为扩展名

utf-8 是一种
文本编码方式

```
<html>  html起始标签
  <head>  head起始标签
    <meta charset="UTF-8">   meta标签：指明文本的编码方式
    <title>美好的一天</title>  一对title标签：
                               中间为网页的标题（名字）
  </head>  head 结束标签
  <body>  body 起始标签
    <h1>计算器和时钟养成记</h1>  h1标签：显示在网页中的文字
  </body>  body结束标签
</html> html结束标签
```

head 元素

body 元素

第一周 提纲

- HTML 结构
- HTML 语法

HTML 语法–概述

- **标记语言：**HTML是一种标记语言，它由一个个标签组成，例如：
  - <html>、</html>
  - <head>、</head>
  - <body>、</body>
- 标签之间允许**嵌套：**即一对标签之间可以定义另一对标签，
  - 上例中，<html>…</html>之间定义了<head>…</head>和<body>…</body>
  - <head>…</head>之间定义了<meta>和<title>
  - <body>…</body>之间定义了<h1>…</h1>
- **与Go不同：**
  - Go作为一种编程语言，常常包含对数字符号的各种操作；
  - 而HTML只是用来指定一个网页上显示什么内容

HTML 语法–基本概念

- **标签:**HTML标签是一个形如"<keyword>"的单词，如<html>
- 大部分HTML标签都按照 <keyword>…</keyword> 的模式
  - 成对出现，如<h1>…</h1>
  - 前面的<h1>称为起始标签（开放标签），后面的</h1>称为结束标签（闭合标签，注意：以 / 开头）
- **属性：**HTML起始标签可以带有属性
  - 属性的值是用引号括起来的字符串，其语法如下：

<h1 *attribute_name*="…">…</h1>

- **元素：**"<key_word>…</key_word>" 这一整体称为一个HTML元素

HTML 语法–<head>中的元素

- 一对<head>…</head>中间一般都需要一个<meta>标签和一对<title>标签，比如：

```
<head>
    <meta charset="UTF-8">
    <title>文本</title>
</head>
```

名字空间：浏览器允许的字符串（中文、英文和空格，多个空格会合并成一个）

- 它们的含义为：

标签	含义
<meta charset="UTF-8">	规定使用 utf-8 编码
<title>…</title>	规定整个网页的标题(名字)

添加小标题–标题

- 王小二觉得要介绍自己的创建过程，他的文章要分为三个部分，取三个标题（显示在网页中），分别为：“发明前”，“发明中”和“发明后”
- 使用以下知识：

- <h1>文本</h1>
- <h2>文本</h2>
- <h3>文本</h3>
- <h4>文本</h4>
- <h5>文本</h5>
- <h6>文本</h6>

字体逐渐变小

代码：

```
<html>
<head>
    <meta charset="UTF-8">
    <title>美好的一天
</title>
</head>
<body>
    <h1>计算器和时钟养成记
</h1>
    <h2>发明前</h2>
    <h2>发明中</h2>
    <h2>发明后</h2>
</body>
</html>
```

效果：

添加自然段-段落、换行、水平线、注释

- 王小二在添加小标题之后，开始兴高采烈地着手写每一个小标题的下面的段落，并且希望每一个小段之间尽可能空开一些。
- 需要用到的知识如下：

段落： 文本

换行：
（或者
）

水平线：<hr/>（或者<hr>）

注释：<!--注释内容-->

代码（只展示body）：

```
<body>
    <h1>计算器和时钟养成记</h1>
    <h2>发明前</h2>
    <p>在发明前，我也不知道自己能不能做成。</p>
    <br>
    <hr>
    <h2>发明中</h2>
    <p>过程中，查了好多资料，好累但很快乐。</p>
    <br>
    <hr>
    <h2>发明后</h2>
    <p>终于做好了，虽然很粗糙，那也是亲儿子</p>
</body>
```

效果：

添加名字和时间–文本格式

- 王小二在添加段落之后，想要在自己的博客上添加名字和书写时间。名字希望斜体加粗并且上标一个小星星，时间希望是斜体。
- 需要用到的知识为文本格式化标签：
  - 文本格式化标签通常是一对标签，这对标签指定了标签中间的文本的格式，例如：

- 加粗：加粗文本
- 斜体：斜体文本
- 上标：^{上标文本}
- 下标：_{下标文本}

代码（只展示body）：

```
<body>
    <h1>计算器和时钟养成记</h1>
    作者：<b><i>王小二</i></b>
<sup>*</sup> 时间：<i>2020年02月26日</i>
    <h2>发明前</h2>
    <p>在发明前，我也不知道自己能不能做成。</p>
    <br>
    <hr>
    <h2>发明中</h2>
    <p>过程中，查了好多资料，好累但很快乐。</p>
    <br>
    <hr>
    <h2>发明后</h2>
    <p>终于做好了，虽然很粗糙，那也是亲儿子</p>
</body>
```

效果：

添加进度表–表格

- 王小二在添加完姓名和时间之后，开始回忆自己的创作过程，希望用一个表格的方式记录自己这几天干的事情。
- 需要用的知识如下：
  - 用一对 **<table>…</table>** 定义**表格**；
  - 在这对标签中间，用*n*对**<tr>…</tr>**定义*n*行；
  - 在每对<tr>…</tr>之间，用*m*对**<td>…</td>**在同一行定义*m*个单元格；
  - 一对<td>…</td>中间的文本就是显示在单元格中的内容。
  - 例如：

```
<table>
        <tr><td>(row0,col0)</td><td>(row0,col1)</td></tr>
        <tr><td>(row1,col0)</td><td>(row1,col1)</td></tr>
</table>
```

代码（“发明中”部分）：

表格框线条粗细

```
<table border="1">
    表：王小二艰难历程表
    <tr>
        <td>事件</td><td>感悟</td>
    </tr>
    <tr>
        <td>看CSS语法</td><td>一天没睡</td>
    </tr>
    <tr>
        <td>看HTML语法</td><td>还是没睡</td>
    </tr>
    <tr>
        <td>看JS语法</td><td>还是GO香</td>
    </tr>
</table>
```

效果：

发明中

过程中，查了好多资料，好累但很快乐。

表：王二小艰难历程表

事件	感悟
看CSS语法	一天没睡
看HTML语法	还是没睡
看JS语法	还是GO香

添加动机–有序列表

把<ol>换成<ul>会定义无序列表（列表项前不显示序号）

- 王小二在添加了创造过程后，突然发现“发明前”还没写，于是用一个有序列表讲了讲自己写网页的目的
  - 有序列表用一对
…
定义，
  - 列表每一项用一对
…
定义，
  - 一对
…
中间的文本是显示在列表项中的内容，
  - 有序列表在浏览器中显示时会自动出现序号

```
<ol>
    <li>文本</li>
    <li>文本</li>
</ol>
```

代码（“发明前”部分）：

```
列表：王小二的动机
<ol>
        <li>学习CSS</li>
        <li>学习HTML</li>
        <li>学习JS</li>
        <li>写计算器和时钟</li>
</ol>
```

效果：

列表：王小二的动机

1. 学习CSS
2. 学习HTML
3. 学习JS
4. 写计算器和时钟

添加计算器–表单

- 王小二在添加了写网页的目的后，想在页面上显示计算器，于是用下面关于表单的知识设计了计算器的样子：

- **表单**是由一对 **<form>…</form>** 组成的元素，在这对标签中间可以使用单一的 **<input>** 定义各种各样的**表单元素**。

- 表单通常用来获取用户的输入，表单内的表单元素通常是文本框或者选项，用户可以在文本框输入文本或者选择选项。

添加计算器–表单元素

- 各种表单元素使用 <input> 上的 **type 属性**来区分：

语法	描述
<input type="text">	能输入文本的框，实时显示用户输入的文本
<input type="password">	用来输入密码的框，将用户输入的字符显示为小圆点
<input type="radio" name="xx">	单选框，同一个表单多个中name属性相同的单选框只能选中一个
<input type="checkbox">	复选框
<input type="button" value="xx">	按钮（value属性的值为按钮上的文字）

代码（“发明后”部分）：

一对 <form>…</form> 中同样**name**属性的单选框是一组，只能选中其一

```
计算器:
  <form>
    操作数1:  <input type="text"><br/>
    操作数2:  <input type="text">
    <br/>
    <input type="radio" name="op">加
    <input type="radio" name="op">减
    <input type="radio" name="op">乘
    <input type="radio" name="op">除
    <input type="radio" name="op">模
    <br/>
    <input type="button" value="计算">
  </form>
```

效果：

- 注：现在只有计算器的样子，还没有定义功能。

HTML语法知识–块级元素 & 内联元素

- 王小二正苦于不知道怎么写，不经意间发现了下面的知识。
  - 在网页中，有的HTML元素可以嵌入在某一行里，比如文本格式化标签<i>组成的元素，具有这种性质的元素称为**内联元素**
  - 反之，像由<h1>元素那样以新的一行开始的元素，称为**块级元素**
- 例子：
  - ** … :**
    - 一个块级元素，无特殊含义。
    - 用于组合多个块级元素（一般网页比较复杂时可为网页分块），方便将来对多个元素进行整体操作。
  - **…:**
    - 一个内联元素，无特殊含义。
    - 用于组合多个行内元素或在某一行内嵌入元素，方便将来操作一行内的一部分。

代码分块– div 标签

- 王小二正好觉得代码有点乱，层次不够清楚，于是使用 标签把代码里发明三阶段分块，这样代码变得好看得多。

```
<body>
    ……
    <div>
        <h2>发明前</h2>
        ……
    </div>
    ……
    <div>
        <h2>发明后</h2>
        ……
    </div>
</body>
```

- 加上 后只是改变了代码的结构，网页并没有什么变化。

给计算器结果留位置– span 标签

- 王小二眉头一皱，发觉计算器没给计算结果留下位置，于是使用 在 <form> 里加上结果的位置。

```
计算器:
<form>
  操作数1:  <input type="text"><br/>
  操作数2:  <input type="text">
  <br/>
  <input type="radio" name="op">加
  <input type="radio" name="op">减
  <input type="radio" name="op">乘
  <input type="radio" name="op">除
  <input type="radio" name="op">模
  <br/>
  结果:  <span>xxxx</span><br/>
  <input type="button" value="计算">
</form>
```

效果：

添加元素的样式

- 王小二改完代码，开始思考怎样让自己的网页更漂亮，于是学习了下面有关CSS的知识：
  - CSS（Cascading Style Sheets，层叠样式表）是一种用来指定HTML元素的样式（如：字体颜色、背景颜色）的语言，其语句具有“名称：值；”的形式，HTML文档有两种方法来加入CSS语句：
- **方法一**：直接在起始标签上定义 **style 属性**，属性的值是若干 CSS 语句，例如：

```
<p style="background-color: gold;">一个金色背景的段落</p>
<p style="color:red;">一个红色字体的段落</p>
```

- 也可以同时规定不同方面的样式

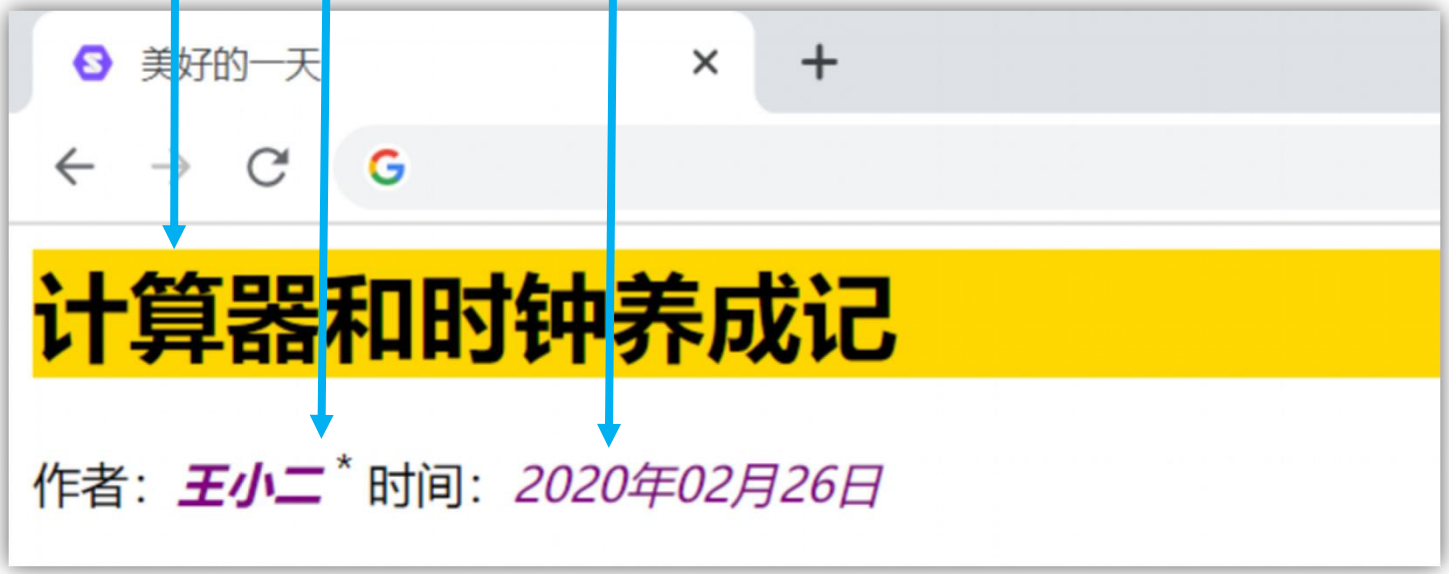
```
<h1 style="color:blue;text-align:center;">一个蓝色字体、居中的标题</h1>
```

注意：CSS 语句中名称和值不能是随便一个字符串，而是 CSS 语法里有的字符串

给元素添加颜色

```
<h1 style="background-color: gold;">计算器和时钟养成记</h1>
作者：<b><i style="color: purple;">王小二</i></b> <sup>*</sup>
时间：<i style="color: purple;">2020年02月26日</i>
```

效果：



添加元素的样式

- 王小二觉得自己的网页还不够好看，想要把发明前、中、后各自美化一下，于是使用了下面的知识：
- **方法二**：在<head>…</head>中间定义**<style>…</style>**，这对<style>内部是若干CSS语句 。
- 例子：

```
<style>
    h1 {color:red;}  /* 规定某类标签的样式：所有<h1>元素字体为红色 */
    #id属性的值 {color:green;}  /* 规定某个元素的样式：被id属性【稍后讲述】指定的元素字体颜色为绿色 */
    .class属性的值 {color:yellow;} /* 规定某“组”元素的样式：具有相同class属性【稍后讲述】的元素字体颜色为黄色 */
</style>
```

元素的通用属性

- 几乎所有的元素都可定义下面几种属性：

属性	描述
id	元素的标识，不同元素需要不同id，可以是任意一个字符串
name	元素的名称，可以是任意一个字符串
class	元素的类，一个元素可以有若干个类，不同元素可以有相同类，可以是任意一个字符串
style	描述了元素的样式

- 定义以上属性的语法

```
<tag_name  attribute1_name="…"  attribute2_name="…" >…</tag_name>
```

给元素添加属性

```
<body>
    ……
    <div id="before">
        ……
    </div>
    <div id="during">
        ……
    </div>
    <div id="after">
        ……
    </div>
</body>
```

- 只是加上id属性不会改变网页的显示效果
- 但是可以被<style></style>中间的语句选中，来添加网页的样式。

给元素添加颜色

和 id 中间不能有空格

```
<style>
    #before {
     background-color:skyblue;
    }
    #during {
     background-color:springgreen;
    }
    #after {
     background-color:tomato;
    }
    h2 {
     background-color:violet;
    }
</style>
```

效果：