# 1.简单机器人编程,其实是训练一下你对函数的理解

## Karel ide: <https://stanford.edu/~cpiech/karel/ide.html>

## 1.使得机器人从左下角移动到右上角

|  |
| --- |
|  |

## 2.使机器人从左下角沿着对角线放置beeper

|  |
| --- |
|  |

## 3.按照棋盘格来放置beeper

|  |  |
| --- | --- |
| function main(){  //your code here  putAndMove();  moveAndPut();  turnLeft();  move();  turnLeft();  movePutMove();  movePutMove();  turnRight();  move();  turnRight();  putAndMove();  moveAndPut();  turnLeft();  move();  turnLeft();  movePutMove();  movePutMove();  turnRight();  move();  turnRight();  putAndMove();  moveAndPut();  }  function putAndMove()  {  putBeeper();  move();  move();  putBeeper();  }  function moveAndPut()  {  move();  move();  putBeeper();  }  function movePutMove()  {  move();  putBeeper();  move();  } |  |

# 2.函数的形参(parameters)和实参(arguments)

## 一个案例,假设我们叫一个机器人去买牛奶,设计一个函数能够计算给定的钱能够卖多少瓶牛奶,并且能够得到多少余额

|  |
| --- |
| <!DOCTYPE html>  <html lang="en">  <head>      <meta charset="UTF-8">      <meta name="viewport" content="width=device-width, initial-scale=1.0">      <title>Document</title>  </head>  <body>      <script>          function getMilkandChange(sum){              console.log("leaveHouse");              console.log("moveRight");              console.log("moveRight");              console.log("moveUp");              console.log("moveUp");              console.log("moveUp");              console.log("moveUp");              console.log("moveRight");              console.log("moveRight");              let cost = 1.5;              let count = Math.floor(sum/1.5);              let change = sum%cost;              console.log(`${sum} dollars can buy ${count} bottles milk (${cost} per bottle)`);              console.log(`and get ${change} dollars change`);                console.log("moveLeft");              console.log("moveLeft");              console.log("moveDown");              console.log("moveDown");              console.log("moveDown");              console.log("moveDown");              console.log("moveLeft");              console.log("moveLeft");              console.log("enterHouse");          }      getMilkandChange(10);      </script>  </body>  </html> |

## 运行效果

|  |
| --- |
|  |

# 3.js random函数,产生一个0-0.9999999999999999的随机数(小数点后面有16位),不过我们可以配合其他函数来得到我们需要的数字,例如获取一个0-6(不包括6)的数字

|  |
| --- |
|  |

## 然后加1,就得到1-7(不包括7)的数字

|  |
| --- |
|  |

## 小案例:利用随机数来测试恋爱指数

|  |
| --- |
| <!DOCTYPE html>  <html lang="en">  <head>      <meta charset="UTF-8">      <meta name="viewport" content="width=device-width, initial-scale=1.0">      <title>Document</title>  </head>  <body>      woman:<input id="woman" type="text"/> ♥ man  <input id="man" type="text"/>      <button id="btn" onclick="showResult()">check</button>      <p id="res" style="font-size: 48px;"></p>      <script>      let woman = document.querySelector("#woman")      let man= document.getElementById("man")      res = document.getElementById("res")      function showResult(){        if(woman.value.length==0 || man.value.length==0)        {          alert("Please Enter Data In All Field! ");          return;        }        else{           let num = Math.random();           num = Math.floor(num \*100)+1;           //alert(woman.value + " Love " + man.value+":"+num+"%");           res.innerHTML=woman.value + " Love " + man.value+":"+num+"%";        }      }      </script>  </body>  </html> |

### 效果

|  |
| --- |
|  |

# 4.==(值等于)和===(全等于)的区别

## 案例:

|  |  |
| --- | --- |
| <script>          function test1()          {              let a = 1;              let b = "1"              if(a===b){                  console.log("Equal");              } else {                  console.log("Not Equal");              }          }          test1();      </script> | function test2()          {              let a = 1;              let b = "1"              if(a==b){                  console.log("Equal");              } else {                  console.log("Not Equal");              }          }          test2(); |
| 结果: Not Equal | 结果: Equal |

### 可见==会做隐式类型转换,===不会,也就是==其实并不安全,也就是==并不关心数据类型,===既要看数值,也要看数据类型.建议使用===来做两个变量的相等判断

# 5.判断是否是润年的函数

|  |
| --- |
| function isLeapYear(year)          {              if(year%4===0)             {               if(year%100===0){                  if(year%400===0){                      console.log(`The Year ${year} is a leap year`);                  } else{                    console.log(`The Year ${year} is not a leap year`);                  }               } else {                console.log(`The Year ${year} is a leap year`);              }             } else {              console.log(`The Year ${year} is not a leap year`);             }          }          isLeapYear(1986); |

## 效果:

|  |
| --- |
|  |

# 判断逻辑是:先看看能否被4整除,如果不能就不是润年,如果能够看看他能否给100整除,如果不能是闰年,如果能够,就看看他能否被400整除如果能够就是闰年,如果不能就不是闰年

# 6.fizzBuzz函数,写一个函数输出从1开始到指定数量的整数,凡是3的倍数输出Fizz,凡是5的倍数输出Buzz,凡是3和5的公倍数,输出FizzBuzz,其他情况输出当前的值

|  |
| --- |
| <!DOCTYPE html>  <html lang="en">  <head>      <meta charset="UTF-8">      <meta name="viewport" content="width=device-width, initial-scale=1.0">      <title>Document</title>  </head>  <body>      <script>          function fizzBuzz(num) {              for(var i=1;i<num;i++)             {                 if(i%3===0 && i%5===0)                {                  console.log("FizzBuzz");                  continue;                }                 else if(i%3===0){                  console.log("Fizz");                  continue;                } else if(i%5===0)                {                  console.log("Buzz");                  continue;                }                console.log(i);             }          }          fizzBuzz(31);      </script>  </body>  </html> |

### 效果:

|  |
| --- |
|  |