

Web 前端课程第二次实验文档

软件 21 周伯威

2012012221

zhou_bw@yeah.net

本文档包括:

基础练习五题 · 进阶练习两题(不含 bonus) · untreated 游戏

基础练习 1:

```
var func={  
  
  getNum:function(){return this.num;},  
  
  num:1  
  
};  
  
(function(){  
  
  return typeof arguments[0]();  
  
})(func.getNum);
```

返回值为 "undefined"。

解释: func 是使用 var 声明的一个对象, 其中的 getNum 与 num 在该语句块内是可见的, 因此 typeof func.getNum 应为 "number"。但是, 在第二个函数中, 传递进去的参数是 func.getNum, 即 "function(){return this.num}", 此处, this 并不是 func, 因此 this.num 实际上是 window.num, 是未定义的, 所以返回结果应为 "undefined"。

基础练习 2:

```
var x=0;

function foo(){

  x++;

  this.x=x;

  return foo;

}

var bar=new new foo;

console.log(bar.x);
```

输出为 undefined。

解释: var bar=new new foo 执行时, 先对 window.x 进行加一操作, 再 this 赋予 this.x=1, 然后函数返回了 foo 这个函数并赋给了 this, 也就是说, 此时返回的是 function foo(){x++;this.x=x;return foo;}。

故 bar.x 为 undefined。

基础练习 3:

```
function bar(){

  return foo;

  foo=10;
```

```
function foo(){}  
  
var foo='11';  
  
}  
  
alert(typeof bar());
```

输出为 `function`。

解释: 我上网查了一下资料。虽然 Javascript 是脚本语言,但在执行前会进行预编译,预编译的过程中, `var` 声明的变量和 `function` 函数会被提前定义。通过实验发现, `var` 与 `function` 的区别是前者只会声明一个变量,而并不赋值,后者则把函数具体内容赋了进去。原题中,在预编译过程中,声明了一个 `function foo`,此时 `foo` 的值就是 `function foo(){}` ,而后又声明了 `var foo`,但因为使用了 `var`,这次并不给 `foo` 赋值, `foo` 为 `undefined`,也就无法覆盖掉原有的定义。故而最后输出为 `function`。

基础练习 4:

```
var x=3;  
  
var foo={  
  
  x:2,  
  
  baz:{  
  
    x:1,  
  
    bar:function(){
```

```
return this.x;}}}  
  
var go=foo.baz.bar;  
  
alert(go());  
  
alert(foo.baz.bar());
```

输出分别为 3 和 1。

解释: 在 var go=foo.baz.bar();这句里,go 的值为 function(){return this.x;}

因此, alert(go());执行时, 相当于执行 alert(this.x), 即 alert(window.x)。

但在执行 alert(foo.baz.bar())时, 相当于执行 alert(foo.baz.x)。

故前者返回 3, 后者返回 1。

基础练习 5:

```
function aaa(){  
  
return {test:1};  
  
}  
  
alert(typeof aaa());
```

输出为 object。

解释: 这段语句相当于 alert(typeof {test:1}); 因为{test:1}是 object, 所以

输出 object。

中国队是冠军!

实现思路: 这是一个数学问题, 某个球队获胜则必须参加四次比赛并均获胜。这四次比赛的手是不确定的, 遇到某个对手都有一定概率, 我们要把这个概率求出来。

我实现了这样一个函数: $g(x,y)$, 其中, x 是要求的国家代号, y 是比赛编号, 如 $g('A1','d2')$ 表示"巴西队进入四强的概率"。它应该等于 $P(\text{"巴西队进入八强"}) \times (P(\text{巴西队胜哥伦比亚})P(\text{哥伦比亚进八强}) + P(\text{巴西胜乌拉圭})P(\text{乌拉圭进八强}))$ 。这样, 最终的 $g(t,'d4')$ 即表示 t 队获得冠军的概率。

在具体实现时, 为了方便找到某队在某比赛中可能遇到的对手, 我使用文本编辑器的列编辑功能, 构建了如下的对象:

```
var r=//x.x.y: x国要进入y比赛之前可能遇到的对手的数组
{
  A1:{d1:["B2"],d2:["C1","D2"],d3:["E1","F2","G1","H2"],d4:["B1","A2","D1","C2","F1","E2","H1","G2"]},
  B2:{d1:["A1"],d2:["C1","D2"],d3:["E1","F2","G1","H2"],d4:["B1","A2","D1","C2","F1","E2","H1","G2"]},
  C1:{d1:["D2"],d2:["A1","B2"],d3:["E1","F2","G1","H2"],d4:["B1","A2","D1","C2","F1","E2","H1","G2"]},
  D2:{d1:["C1"],d2:["A1","B2"],d3:["E1","F2","G1","H2"],d4:["B1","A2","D1","C2","F1","E2","H1","G2"]},
  E1:{d1:["F2"],d2:["G1","H2"],d3:["A1","B2","C1","D2"],d4:["B1","A2","D1","C2","F1","E2","H1","G2"]},
  F2:{d1:["E1"],d2:["G1","H2"],d3:["A1","B2","C1","D2"],d4:["B1","A2","D1","C2","F1","E2","H1","G2"]},
  G1:{d1:["H2"],d2:["E1","F2"],d3:["A1","B2","C1","D2"],d4:["B1","A2","D1","C2","F1","E2","H1","G2"]},
  H2:{d1:["G1"],d2:["E1","F2"],d3:["A1","B2","C1","D2"],d4:["B1","A2","D1","C2","F1","E2","H1","G2"]},
  A2:{d1:["B1"],d2:["C2","D1"],d3:["F1","E2","H1","G2"],d4:["A1","B2","C1","D2","E1","F2","G1","H2"]},
  B1:{d1:["A2"],d2:["C2","D1"],d3:["F1","E2","H1","G2"],d4:["A1","B2","C1","D2","E1","F2","G1","H2"]},
  C2:{d1:["D1"],d2:["A2","B1"],d3:["F1","E2","H1","G2"],d4:["A1","B2","C1","D2","E1","F2","G1","H2"]},
  D1:{d1:["C2"],d2:["A2","B1"],d3:["F1","E2","H1","G2"],d4:["A1","B2","C1","D2","E1","F2","G1","H2"]},
  E2:{d1:["F1"],d2:["G2","H1"],d3:["B1","A2","D1","C2"],d4:["A1","B2","C1","D2","E1","F2","G1","H2"]},
  F1:{d1:["E2"],d2:["G2","H1"],d3:["B1","A2","D1","C2"],d4:["A1","B2","C1","D2","E1","F2","G1","H2"]},
  G2:{d1:["H1"],d2:["E2","F1"],d3:["B1","A2","D1","C2"],d4:["A1","B2","C1","D2","E1","F2","G1","H2"]},
  H1:{d1:["G2"],d2:["E2","F1"],d3:["B1","A2","D1","C2"],d4:["A1","B2","C1","D2","E1","F2","G1","H2"]}
};
```

这样便可轻松地利用循环语句计算概率。

找呀找呀找同学

实现思路: 对于输入的第二个参数 r , 我使用 `switch` 与 `typeof` 来判断

其数据类型。根据 r 的数据类型，在输入数据中逐一查找符合要求的数据，如符合要求，将之 push 入 ans 数组，最后再根据要求做对应输出。

Untrusted

level 1:

<https://gist.github.com/d2da0e3e040572da2de2>

把代码删掉，墙就没了

level 2:

<https://gist.github.com/d9c3873b7c7498ef8429>

把□放在小@旁边

level 3:

<https://gist.github.com/04e376d43f31e9a621cb>

把墙移动一下

level 4:

<https://gist.github.com/38973461fbc8009b8bde>

把□放在墙内

level 5:

<https://gist.github.com/82edbf6b5bc9266e9eda>

把 75 个墙堆到左上角，然后 break 掉随机生成墙的循环

level 6:

<https://gist.github.com/6e42718d1edc7caf3543>

安两个墙把□挡住

level 7:

<https://gist.github.com/9dab2ef9671a452ed2d9>

在  的前面改变@的颜色

level 8:

<https://gist.github.com/3af7d3c78c864cb2ee79>

一直随机生成地图

level ⑨

<https://gist.github.com/588bac20558e7cfe6163>

在水面上噗满小船，小船是我自己从 raft 那 copy 的 Object，名字叫
"haha"。。。

level 10:

<https://gist.github.com/650b9194b1066cbfde5b>

不太好形容，像这样让个道出来。。



level 11:

<https://gist.github.com/f492bc50e76691c9efdd>

控制机器人: 尽量向右移动，撞到墙后向下移动

level 12:

<https://gist.github.com/ec0f363e2737beac8e0a>

通过 getX、getY 判断机器人位置，让他绕过障碍

level 13:

<https://gist.github.com/d4d6d8e0cb2ebaf8d6a7>

策略: 让机器人尽量往下走，碰到墙就往右走，然后不断刷新，碰运气刷到如下图这样一个可以走的



level 14:

<https://gist.github.com/56958f00f9f4a56d48ba>

把蓝钥匙扔掉

level 15:

<https://gist.github.com/bb134e30e7a8b24bc7ea>

```
player.killedby(a);
```

于是跳到水里就提示 a 未定义什么的，然后就死不掉了==

level 16:

<https://gist.github.com/abd9c020a06d0effc525>

先让 player 变成黄颜色，然后画线的时候把颜色都画出来。。

level 17:

<https://gist.github.com/ab1367e4b979245da2ad>

把所有传送到传送门的传送门都标记出来了

level 18:

<https://gist.github.com/3e475c067602e38a4d01>

如图。搭桥，桥是我自己弄的方块构成的，否则会提示 block 太多



level 19:

<https://gist.github.com/e33c9642fd14d54253fc>

乱按按过去的。。

level 20:

<https://gist.github.com/8f9629e2530023331a72>

我比较擅长弹幕游戏嗯，先是直接走过去发现不行要打 boss

然后试着往 boss 上加 item，发现会被 boss 吃掉

然后试着加 dynamic，发现对它数量有限制

然后嗯。。用了 overrideKey，重新定义了左键，从左边发射子弹打

boss。。

