# 测试计划

**序号：13**

## 1概述

### 1.1系统的简介

FlappyBird

Flappy bird is a form of simple but very casual games. Players need to keep control of the frequency of the click on the screen to adjust the flying height of the birds and landing speed, and let the bird pipe aperture pass the picture to the right. If the bird accidentally brush against the pipe, the game has ended. This design introduced the computer version of the game about framework and specific source program code.

### 1.2目的和范围

测试游戏的Bug

## 2测试过程

### 2.1测试的目标

(包括测试的目的，测试的类型，测试的级别)

目的：通过测试对象进行的浏览可正确反映游戏是否成功运行

类型：手工测试

级别：标准

### 2.2测试的指导方针

各项测试活动的分工（例如谁负责生成用于测试的可执行代码，谁来负责具体执行各项测试）；测试的步骤；测试过程需要用到的各种检查表、报表等；测试过程中配置管理步骤。

集成测试的目的是确保程序满足概要设计说明书的要求。它所测试的内容包括模块的功能，模块间的接口以及集成后的功能，并且对以前集成的build进行增量式测试。

例

本次一共分三轮测试。使用交叉测试法、因果关系法、等价划分法和约束法。 第一轮测试  开发完一个模块，就测一个模块。

以功能测试为主，同时兼顾用户界面测试，易用性测试，数据和数据库完整性

测试及性能测试。尽可能将存在的问题暴露出来。

确保业务流程能走通，尽可能将需求中的功能点核实。 所设计的测试用例都执行完。并补充相应的测试用例。

第二轮测试   保证系统正常功能正确的情况下对边界和一些特殊的情况。

保证系统界面符合界面规范和友好性符合用户操作习惯。

保证多用户并发操作时模块功能实现正确。 系统中所有功能按正常流程都能正确实现。 在规定的测试时间段内按要求完成测试。

经过测试保证系统符合项目规范；网页可读性；网页下载速度；系统便用性；浏览器兼容性等多个方面。

核实测试对象可在所需的硬件和软件配置中正常运行。

### 2.3评价标准

用于判定测试用例通过与否的指导原则（例如，执行完毕并且没有系统错误；满足性能要求；产生了预期的结果等等），以及具体的评分标准。

1) 测试用例覆盖率, 包括功能覆盖和非功能覆盖, 这两个覆盖率都需要达到100%

2) 测试用例涵盖的范围, 包括功能测试和非功能测试, 一个完整的测试用例, 应该包含这2个部分.

3) 测试用例描述的正确性, 测试用例描述的清楚明了, 不模棱两可.

4) 有充分的负面测试, 作为测试用例，不仅要测试正确的输入和操作，还要测试各种各样的例外情况，如边界条件、不正确的操作、错误的数据输入等

5) 测试用例的单一性, 一个测试用例面向一个测试点，不要将许多测试点揉在一起。例如，通过一个测试用例发现1～2个缺陷，而不是发现5～10个缺陷甚至更多的缺陷。

6) 符合测试用例设计规范或测试用例模版. 一个规范的测试用例, 应该是遵循测试用例设计规范或测试用例模版的, 有固定的格式.

7) 操作步骤的准确性, 按照测试用例的步骤操作, 可以得到唯一的测试结果

8) 操作步骤的简单性, 操作步骤不宜太复杂, 过于复杂的操作步骤意味着测试用例需要被分解为多个测试用例或者分解为多个环节进行验证。

9) 测试环境的正确性, 测试数据的充分性, 测试用例中提到的测试环境是可搭建的, 测试数据要准备充分.

10) 每百个用例发现的缺陷数, 每100个测试用例中发现的缺陷数目

测试用例质量=缺陷总数/用例总数\*100=\*\*（个/百用例）

### 2.4纠正错误和重新测试的步骤

1） 基本流和备选流

采用此方法进行设计时，需要进行场景的设计，在场景中采用基本流和备选流表示经过用例的每条路径

基本流：采用直黑线表示，是经过用例的最简单的路径（无任何差错，程序从开始直接执行到结束）

备选流：采用不同颜色表示，一个备选流可能从基本流开始，在某个特定条件下执行，然后重新加入基本流中，也可以起源于另一个备选流，或终止用例，不在加入到基本流中；（各种错误情况）

（2） 设计步骤

根据说明，描述出程序的基本流及各项备选流

根据基本流和各项备选流生成不同的场景

对每一个场景生成相应的测试用例

对生成的所有测试用例重新复审，去掉多余的测试用例，测试用例确定后，对每一个测试用例确定测试数据值

三．黑盒测试用例设计方法的选择策略

1．首先进行等价类划分，包括输入条件和输出条件的等价类划分，将无限测试变成有限测试，这是减少测试量和提高测试效率的最有效办法

2．在任何情况下都必须使用边界值分析方法。此方法设计的测试用例发现程序错误的能力最强

3．可以用错误和推测法追加一些测试用例

4．对照程序的逻辑，检查已设计的测试用例的逻辑覆盖度，如果没有达到要求，应在补充

5．如果程序的功能说明中含有输入条件的组合情况，一开始就可以使用因果图法和判定表驱动法

6．对于参数配置类的软件，要用正交试验法选择较少的组合方式达到最佳效果

7．功能图法也是很好的测试用例设计方法，我们可以通过不同时期条件的有效性设计不同的数据

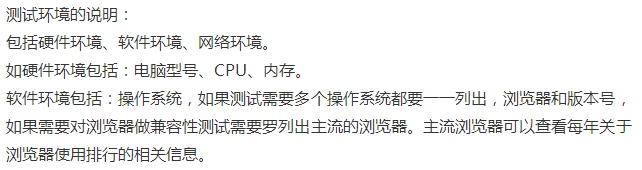
8．对于业务流清晰的系统，可以利用场景法贯空整个测试案例过程，在案例中综合使用各种方法

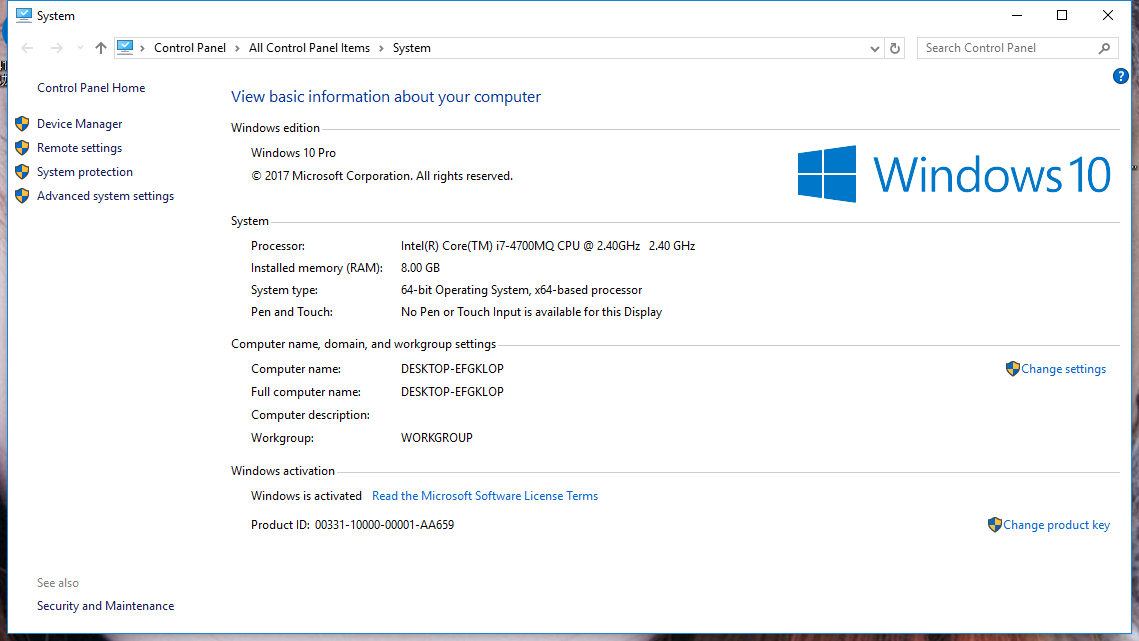
## 3测试小结

### 3.1测试环境要求

外部接口的设置；准备必需的资源。

Windows 10系统





### 3.2简要描述准备执行的各个测试用例

1、测试用例名称，也叫测试用例标题，一定要写得简洁、明了，需要用概括的语言描述该用例的出发点和关注点，使得测试人员第一眼看到测试用例名称就能够明白测试用例的目的。用例名称中一般要求不能存在假设性的语句，并且原则上每个用例的名称不能重复。

2、预置条件要明确，包括测试环境、测试数据、测试场景。因为许多BUG只有在特定的环境、特定的场景下才可以重现。没有正确的前提条件，就无法进行后面的测试步骤或无法得到预期的结果。

3、测试步骤描述要简单、清晰，并且要清楚每一个步骤的描述，比如：第一步，输入用户姓名；第二步，输入登录密码；第三步，用户点击登录。步骤写的明确时就利于提高用例的可操作性。

4、用例的预期结果要完整而且清晰，并且要将各个输出的结果写出来，包括：返回值的内容、数据库相关字段的记录、界面的响应结果、输出结果的规则符合度、日志的检查和对其它业务影响的检查。

5、测试用例级别要划分清楚，这样在测试执行时有主次之分。

6、测试用例的划分也要单一，一个测试用例只检查功能点的一种情况。一个用例检查的情况太多，会导致用例的目的不明确。而且这样组织用例，有利于需求覆盖率的统计。一个功能点我们测试了哪些情况，以及哪些功能点我们在重点测试，一目了然。

### 3.3需求跟踪

(创建一个跟踪矩阵来描述功能需求与测试用例之间的关系。)

