命令行计算器

测试计划

版本 <2.0>

修订历史记录

|  |  |  |  |
| --- | --- | --- | --- |
| **日期** | **版本** | **说明** | **作者** |
| 2019/4/13 | 1.0 | 增加测试计划 | 吴志文 |
| 2019/4/13 | 1.1 | 补全测试种类 | 姜凡叙 |
| 2019/5/12 | 2.0 | 增加单元测试 | 吴志文 |
|  |  |  |  |

目录

1. 简介 4

1.1 目的 4

1.2 背景 4

2. 测试需求 5

3. 测试策略 6

3.1 测试用例 6

3.1.1 等价类划分并测试 6

3.1.2 边界值分析 6

3.1.3 复杂表示式计算 7

3.1.4 无效输入 7

3.1.5 功能测试 8

3.1.6 单元测试 9

测试计划

# 简介

## 目的

命令行计算器 的这一“测试计划”文档有助于实现以下目标：

利用等价类划分和边界值测试设计测试用例，并对命令行计算器进行黑盒测试。通过黑盒测试来验证命令行计算器的主要功能，包括但不限于加减乘除、阶乘、幂指运算、取余等

黑盒测试:也称功能测试，它是通过测试来检测每个功能是否都能正常使用。在测 试地，把程序看作一个不能打开的黑盒子，在完全不考虑程序内部结构和内部特性的 情况下，在程序接口进行测试，它只检查程序功能是否按照需求规格说明书的规定正 常使用，程序是否能适当地接收输入数据而产生正确的输出信息。黑盒测试着眼于程 序外部结构，不考虑内部逻辑结构，主要针对软件界面和软件功能进行测试。黑盒测 试是以用户的角度，从输入数据与输出数据的对应关系出发进行测试的。很明显，如 果外部特性本身有问题或规格说明的规定有误，用墨盒测试方法是发现不了的。

等价类是指某个输入域的子集合。在该子集合中，各个输入数据对于揭露程序中的错误都是等效的，具有等价特性。等价类划分可有两种不同的情况：有效等价类和无效等价类。有效等价类是指对于程序的需求规格说明来说是合理的，有意义的输入数据构成的集合。利用有效等价类可检验程序是否实现了规格说明中所规定的功能和性能（确认过程）。无效等价类是指是指对于程序的需求规格说明来说是不合理的，无意义的输入数据构成的集合。利用无效等价类可检验程序对于无效数据的异常处理能力（检验过程）。

边界值分析法就是对输入或输出的边界值进行测试的一种黑盒测试方法。通常边界值分析法是作为对等价类划分法的补充，这种情况下，其测试用例通常来自等价类的边界。

利用白盒测试等结构测试方法，对命令行计算器进行单元测试，来验证命令行计算器的各个函数的功能和正确性，包括Power,FindNumber,DealWithNegative,Init,Mod,DelSpace,Factorial等等。

白盒测试目的：通过检查软件内部的逻辑结构，对软件中的逻辑路径进行覆盖测试；在程序不同地方设立检查点，检查程序的状态，以确定实际运行状态与预期状态是否一致

单元测试：指对软件中的最小可测试单元进行检查和验证。在本次测试中，我们对于命令行计算器的各个重要的功能函数进行了测试。

## 背景

测试命令行计算器主要功能，包括加减乘除、阶乘、幂指运算、取余等。

# 测试需求

下面列出了那些已被确定为测试对象的项目（用例、功能性需求和非功能性需求）。此列表说明了测试的对象。

|  |  |
| --- | --- |
| 测试项目名称 | 命令行计算器 |
| 功能需求 | 测试加法、减法、乘法、除法、阶乘、幂指运算、取余 |
| 单元测试 | Power,FindNumber,DealWithNegative,Init,Mod,DelSpace,Factorial等 |

# 测试策略

## 黑盒测试

### 等价类划分并测试

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 等价类 | 加 | 减 | 乘 | 除 | 阶乘 | 幂运算 | 取余数 | 预期输出 |
| 整型 | 30+40 | 70-49 | 15\*16 | 36/4 | 5! | 5^2 | 7%3 | 正常运算 |
| 小数 | 15.2+ 30.4 | 20.6-30.23 | 5.5\*3.4 | 10.2/5.3 | (3.4)! | 3.2^3.3 | 15.5%2.3 | 除阶乘和取余外正常运算 |
| 负数 | -3+-4 | (-4)-(-5) | -12\*-13 | -20/ -4 | (-5)! | (-5)^(-1) | -29%-11 | 除阶乘外正常运算 |
| 无效输入 | 2e+3 | 3e-4 | E\*2 | e/2 | E! | E^2 | E%5 | 输出错误信息 |

### 边界值分析

|  |  |  |  |
| --- | --- | --- | --- |
| 测试用例编号 | 被加数 | 加数 | 预期输出 |
| 1 | 10.5 | 10 | 正常运算 |
| 2 | 0 | 10.5 | 正常运算 |
| 3 | 70 | 0 | 正常运算 |
| 4 | -75 | 10 | 正常运算 |
| 5 | 100 | -10 | 正常运算 |
| 6 | 101 | -10.5 | 正常运算 |

|  |  |  |  |
| --- | --- | --- | --- |
| 测试用例编号 | 被减数 | 减数 | 预期输出 |
| 7 | 10.5 | 10 | 正常运算 |
| 8 | 0 | 10.5 | 正常运算 |
| 9 | 70 | 0 | 正常运算 |
| 10 | -75 | 10 | 正常运算 |
| 11 | 100 | -10 | 正常运算 |
| 12 | 101 | -10.5 | 正常运算 |

|  |  |  |  |
| --- | --- | --- | --- |
| 测试用例编号 | 被乘数 | 乘数 | 预期输出 |
| 13 | 10.5 | 10 | 正常运算 |
| 14 | 0 | 10.5 | 正常运算 |
| 15 | 70 | 0 | 正常运算 |
| 16 | -75 | 10 | 正常运算 |
| 17 | 100 | -10 | 正常运算 |
| 18 | 101 | -10.5 | 正常运算 |

|  |  |  |  |
| --- | --- | --- | --- |
| 测试用例编号 | 被除数 | 除数 | 预期输出 |
| 19 | 10.5 | 10 | 正常运算 |
| 20 | 0 | 10.5 | 正常运算 |
| 21 | 70 | 0 | **除零错误** |
| 22 | -75 | 10 | 正常运算 |
| 23 | 100 | -10 | 正常运算 |
| 24 | 101 | -10.5 | 正常运算 |

|  |  |  |  |
| --- | --- | --- | --- |
| 测试用例编号 | 底数 | 指数 | 预期输出 |
| 25 | 10.5 | 10 | 正常运算 |
| 26 | 0 | 10.5 | 正常运算 |
| 27 | 70 | 0 | 正常运算 |
| 28 | -75 | 10 | 正常运算 |
| 29 | 100 | -10 | 正常运算 |
| 30 | 101 | -10.5 | 正常运算 |

|  |  |  |
| --- | --- | --- |
| 测试用例编号 | 阶乘数 | 预期输出 |
| 31 | 10.5 | **报错信息** |
| 32 | 0 | 正常运算 |
| 33 | 10 | 正常运算 |
| 34 | -10 | **报错信息** |

|  |  |  |  |
| --- | --- | --- | --- |
| 测试用例编号 | 被除数（取余） | 除数 | 预期输出 |
| 35 | 10.5 | 10 | **报错信息** |
| 36 | 0 | 10.5 | **报错信息** |
| 37 | 70 | 0 | **除零错误** |
| 38 | -75 | 10 | 正常运算 |
| 39 | 100 | -10 | 正常运算 |
| 40 | 0 | -7 | 正常运算 |

### 复杂表示式计算

|  |  |  |
| --- | --- | --- |
| 测试用例编号 | 表达式 | 预期输出 |
| 41 | 3+++4--3 | 10 |
| 42 | 3\*(2+3)/0 | 除零错误 |
| 43 | 3+4^2\*4 | 67 |
| 44 | 4/2^2\*2 | 2 |
| 45 | 36/3!^2 | 1 |
| 46 | -10^2/2+-3 | -53 |
| 47 | 1.6^2/2! | 1.28 |
| 48 | 1-2\*(30+(-40.0/5)\*(19-4!/2))-(-4\*3)/2^(8-3\*2) | 56 |
| 49 | 2+3M;  R\*5 | 25 |
| 50 | 3-3M;  9%R | 除零错误 |

### 无效输入

|  |  |  |
| --- | --- | --- |
| 测试用例编号 | 表达式 | 预期输出 |
| 1 | 3/ | **报错信息** |
| 2 | \*\*5 | **报错信息** |
| 3 | 5/\*7 | **报错信息** |
| 4 | 1…3\*5 | **报错信息** |
| 5 | <空> | **报错信息** |
| 6 | <未设置M时>R+1 | **报错信息** |
| 7 | 5\*(6+(3-1) | **报错信息** |
| 8 | !7 | **报错信息** |

## 单元测试

### 功能测试

|  |  |
| --- | --- |
| 测试目标： | 确保程序能正确计算加减乘除、阶乘、幂指、取余  确保程序能正确应对用户输入无效数据 |
| 方法： | 在命令行输入测试用例要求的输入，记录测试结果 |
| 完成标准： | 对于正确的输入，正确得到计算结果  对于无效的输入，输出错误提示信息 |
| 需考虑的特殊事项： | 计算时，可能大整数会溢出  由于计算机本身原因，可能会出现精度失真 |

### 3.2.2 测试用例

|  |  |  |  |
| --- | --- | --- | --- |
| 测试目标 | 功能 | 方法 | 预期结果 |
| StrToNum | 将字符串变成数字，不为纯数字则报错 | 分支覆盖 | 通过测试 |
| LessPrimary | 处理含有^的字符串 | 分支覆盖 | 通过测试 |
| GetToken | 在一个字符串中返回和给定类型的op相似token的位置 | 分支覆盖 | 通过测试 |
| Mod | 模运算 | 语句覆盖 | 通过测试 |
| Power | 幂运算 | 语句覆盖 | 通过测试 |
| Init | 处理一个字符串连续的加减或M、R，给出最简 | 分支覆盖 | 通过测试 |
| Term | 只处理乘或除或模字符串的 值 | 分支覆盖 | 通过测试 |
| FindNumber | 在一个字符串中返回从给定位置之后的第一个非数字非dot的位置 | 语句覆盖 | 通过测试 |
| Primary | 处理含有！的字符串 | 分支覆盖 | 通过测试 |
| Factorial | 阶乘运算 | 分支覆盖 | 通过测试 |
| Calculate | 计算器主入口，处理初始字符串 | 分支覆盖 | 通过测试 |
| VeryPriamry | 处理一个字符串的（）中的数字或expression | 分支覆盖 | 通过测试 |
| DelSpace | 对字符串删除空格和tab | 语句覆盖 | 通过测试 |
| DealWithNegative | 对有+或-在前的数字的字符串，加上（） | 语句覆盖 | 通过测试 |
| Expression | 处理含有加或减的字符串的值 | 分支覆盖 | 通过测试 |