ソフトウェア開発環境における人的エラーと人的要因に関する研究

EKUBO, Naoaki / 鬠, 尚晃

(発行年 / Year)
2013-03-24
(学位授与年月日 / Date of Granted)
2013-03-24
(学位名 / Degree Name)
修士(工学)
(学位授与機関 / Degree Grantor)
法政大学 (Hosei University)

2012 年度　修士論文

ソフトウェア開発環境における人的エラーと人的要因に関する研究

A STUDY ON A RELATIONSHIP BETWEEN HUMAN ERRORS AND HUMAN FACTORS IN SOFTWARE DEVELOPMENT ENVIRONMENT

法政大学大学院工学研究科

システム工学専攻修士課程

11R6205　　酈　尚晃

Naoaki Ekubo

指導教員　木村光宏

要旨

ソフトウェアの開発工程において生じる人為的誤りは，ソフトウェアの信頼性に多大な影響を与える．また，フォールトの発生（プログラム内への無意識による埋込み）に影響を及ぼす人的要因を解明できればソフトウェアの信頼性向上技術の開発や改善に期待できる．

そこで本研究では，ソフトウェア信頼性に影響を及ぼす人的要因の分析を行う為に，人的要因を設定した上で，ソフトウェア開発環境におけるコードレビュー実験を行う．コードレビュー実験によって得られたデータを基に 2 通りの手法を用いて人的エラーの発生原因となる人的要因を考察する．1 つは，実験計画法の直交表を基づく分散分析を行い，フォールトを見逃す原因となった人的要因や，それらの交互作用を明らかにする．2 つめは，共分散分析を行い，人的要因の見逃しがなかったかなどについて検討する．

Abstract

In order to effectively reduce the number of software faults in the software development phase, we focus on human factors on the code review process. In this study, we try to find the human factors which affect the efficiency of code review activities by using several experiments. Especially, we use the so-called design of experiment to find the significant factors, and also apply the analysis of covariance to the data sets.

目次

1. まえがき

現在，情報社会が進化しており，それに伴ってソフトウェア製品や，システムの信頼性は益々向上しなければならないと言える．その為にはソフトウェアの開発環境がより優れて発展して行かねばならない．また，一般に情報システムは，現在社会においてはなくてはならない重要な存在になっている．そうしたシステムが誤作動を起こしてしまうと，被害が計り知れないものとなってしまう．そのためにも，ソフトウェアの開発環境がより優れて発展して行かねばならない．また，ソフトウェア開発工程において様々なエラーが発生しているが，その中でも人的誤りが引き起こすエラーはソフトウェア開発に多大な影響を及ぼしている．そうしたソフトウェア開発時の人の誤りや，欠陥などの人的エラーは多種多様な人的要因が相互に関連しあって引き起こされるとも考えられる．また，ソフトウェア開発を行う際に，人が引き起こす原因によりバグや欠陥などがプログラム内に作り込まれ，それらはソフトウェアの信頼性を脅かす問題点になっている．そこで，ソフトウェア製品の品質や信頼性を向上する為には，これらの影響分子を判明させなければならない．こうした人的要因を解明している研究は数多く行われている．しかし，その要因を数値化し解明しているものは少ない．本研究では，以下の点について検証を行う．

- 人的要因について考察するのではなく，人的要因の交互作用についても考察する．

- 実際のソフトウェア開発実験を行い，データを収集して，それを基に人的要因を考察する．

これまでに，人的要因に関する研究はいくつもおこなわれているが[1，2，3]，その交互作用の関連性を明らかにしているものは少ない．また，その要因の一部を数値化し取り扱うことで，フォールト発見数と数値との比較を行うことができる．本研究では，ソフトウェア開発実験を行い，人的要因の分析を行う．始めに人的要因と実験モデルを設定した上で，ソフトウェア開発環境を想

定したコードレビュー実験を行う．そうして得られた実験データから実験計画法の直交表を適用した分散分析を行う．

本研究では第 2 章で既存の研究について説明をした後，第 3 章では人的要因とその交互作用を分析するために，エラーの原因が直接的な原因である「素因」と間接的原因である「誘因」という 2 種類の要因からなる人的要因モデルを設定する．次に人的要因の影響が大きいと考えられるコードレビュー工程に対する実験モデルを仮定する．実験に際して，直交表による実験計画法を使ったデータ分析を行う．これは品質工学等の分野でよく使われ，目的とする特性に対して影響する多数の要因効果とその相互関係を同時に評価するができる．これにより，多数の人的要因の影響を同時に評価し，人的要因の相互関係を分析する．得られたデータを基に 2 通りの方法で分析を行う．

第 4 章では，第一の手法である分散分析を行う．これは人的要因を直交表に当てはめて，分散分析を行い，人的エラーの原因となった人的要因とその交互作用を明らかにする．

第 5 章では数値化した要因を共変量とおいて，共分散分析を行う．第 4 章で見逃していた要因を検証する．

最後に第 6 章では，本研究における実験及び分析結果を考察し論ずる．

2. 既存の研究

ここでは，文献[1]で行われていた実験とその結果について述べる．既存の研究ではプログラムに前もって論理エラーと文法エラーを実験者が人為的に挿入し，それを被験者によって発見する実験を行っていた．この時に使われ人的な要因としては，『言語能力の有無』,『チェックシートの有無』,『コードレビュー時間の長短』,『エラーの摘出目標値の有無』であった．結果として，人的要因としては言語能力の有無が要因として大きな影響を及ぼしていることが示された．本研究では，それ以外の要因をも模索することを意図して，また言語能力の有無を採用せず，それとは違うアプローチで実験を行いたいと考えた．

2.1 人的要因モデル

人的要因が多種多様に関わる事が，エラーの作りこみの原因になっていることについてはほぼ間違いないと思われる．前述した既存の研究[1]では，人的要因モデルは，二つの要因から成るとしている．

一つ目は内部的要因である素因．二つ目は外部的要因である誘因に分かれる，素因とは人間の内部にあって行動に影響を与えるものである．ここでは作業に誤りを起こさせ，エラーの直接的原因になるものを指す．例をあげると，言語能力，思考能力，作業意欲などが挙げられる．一方，誘因とは，その逆であり，人間の外部から干渉するもので間接的に影響が与えられるものである．一般的には，作業に誤りを誘発させてしまう可能性があるものであるとされる．例を挙げると，開発環境，作業時間，騒音などが挙げられる．この二つを合わせてものが人的要因となる．この人的要因モデルから実験モデルを設定し，実験を行っていく．

2.2 実験内容

文献[1]では，人的要因のみでは無くその相互関係(交互作用)を検証している．また，実際にソフトウェアで使われているコードレビューに関する実験を行なってデータを収集して，検証している．ここでコードレビューとは，ソースコードの正しさを検証す

るために被験者がソースコードを机上にてレビューして見直すことである．コードレビュー実験とは，ソースコードレビューを想定した実験を指すが，ソースコード（プログラム）内に実験実施者が予めいくつかの誤りを埋め込んでおき被験者がそれらのうちの何割かを事前に指摘することである．文献[1]では，実験に用いたプログラムは数値の大小比較のアルゴリズムを Fortran 言語で実装したものであり，何らかの専門的知識に依存するような特別に難しい内容にしていなかった．また，エラーの埋め込み個数は 10 個で，論理エラーを 7 個，文法エラーを 3 個埋め込んである．被験者は Fortran 言語の教育を受けた 38 名の学生から成り，素因を言語能力の高低，誘因をチェックリストの有無，レビュー時間，エラー摘出目標値の有無を仮定し実験を行った．これらの人的要因を直行表に割り当てて，分散分析を行った結果，エラーの指摘数の大小に影響を与える要素として言語能力，摘出目標値はエラー発見に有意であるとわかり，チェックリスト，レビュー時間については効果がなかった．因子の交互作用で効果があったのは「言語能力が高く，チェックリストがない」,「チェックリストがなく，摘出目標値がある」であった．このことから最適水準は「言語能力があり，チェックリストがなく，摘出目標値がある」となった． このことから，この実験では言語能力の有無は被験者の自己申告によっている為，この点は本研究において何らかの他の素因で置き換えることを検討することとした．

3. 本研究での実験

3.1 実験モデル

本研究での実験モデルを作る際，ソフトウェア開発における局面を想定して作らなければならない．また，開発工程によって，人的エラーは異なっている．したがって，工程によって対応する人的要因が異なっている．そこで人的エラーが大きい工程を対象にして実験モデルを設定し，分析を行い，ソフトウェアの信頼性向上や改善の為の要因を探る．人的要因が起こりやすい工程として設計とレビュー工程がある．今回は，実験対象を学生としている為，設計の知識がない．その為，レビュー工程の方が実験が容易であり，正しい結果が出るであろうとし，コードレビュー工程に設定した．コードレビューとはコーティングを終えたプログラムを実施前に机上で解析して，誤りやフォールトを発見して修正することである．

今回の実験では人の手による作業が多い為，エラー（バグを指摘できないこと）に対する人的影響が大きいと考えられる．本実験ではソースプログラムに既知の論理エラーを実験者が埋め込み，被験者に指摘してもらう実験を行う．要因ごとに場合分けをした被験者がエラーを見逃したり誤って指摘したりすることを人的エラーとみなす．問題のプログラムは問題分野の知識に強く依存するような内容や，複雑な言語機能を用いなければ記述できないような内容でなく，一般的に知識で解ける「数字当てゲーム」と「三角形の判別の」の 2 種類の問題に設定した．その中でも文法エラーではなく論理エラーを入れ込んだ実験を行うこととした．なぜなら文法エラーはコンパイルなどで削除が可能であるからである．

3.2 実験内容

(1)素因と誘因の仮定

実験内容としてプログラムに既知の論理エラーを数個埋め込み，被験者がレビューしてエラーを取り除く方法で行う．実験を行う際の素因と誘因は以下の様に設定した．

A) CAB 適正検査の得点高低

SHL 社で作られている一般企業での採用テストの一つ[4]で，主にコンピュータ職に採用されるテストとして使われている．これらの試験を被験者にあらかじめ行うことで，能力の高低を分けておく．これを因子 A とおき，高を第 1 水準，低を第 2 水準とした．なお，その結果は被験者には告知はしていない．

B) BGM の有り無し

BGM として人間の脳を落ち着かせる効果があると思われるヨハン・パッヘルベルの「カノン」を採用した．これを因子 B とおき，音有りを第 1 水準，音無しを第 2 水準とした．

C) コードレビュー時間の長短

文献[1]を参考にして,コードレビュー時間を 15 分と 25 分の二つに分ける．これを因子 C とおき，25 分を第 1 水準，15 分を第 2 水準とした．

(2)被験者の設定

実験をしてもらう被験者を本学理工学部の 3 年生・4 年生・院生とする．ここでいずれも C 言語のプログラムの経験がある生徒を対象とした．また，被験者数は各水準に対して 3 人とした．

(3)問題プログラムの選定

問題プログラムは 2 つにした．1 つは,「三角形の判別」．もう 1 つは,「数字当てゲーム」．いずれも専門的な知識がいらない，一般知識で解ける問題とした．使った言語は，C 言語である．

(4)フォールトの埋め込み内容とフォールトの内容

フォールトの埋込みは，数値当てゲームに 6 個，三角形の判別に 4 個の計 10 個埋め込んだ．いずれも構文エラー，文法エラーなどのコンパイルで検出できるエラーは含まれない．

(5)データの収集方法

データの収集は，被験者に CAB のテストをしてもらい，それぞれ高い能力と低い能力に分けた．さらに BGM の有り無し，レビュー時間を 15 分と 25 分に分けて 8 通りの条件に分けてデータ

を収集する．

3.3　直交表への人的要因の割り付け

多数因子の実験では，全ての水準の組み合わせを使い実験をするので単一因子実験より高度な精度のデータが取れる．しかし，因子の数が多くなると水準の組み合わせが多くなり，実験のデータを取るのが難しくなる．

例えば因子数が 10 個あれば，すべてが 2 水準でも水準少なくとも 1024 回の実験が必要になる．これは実施が難しい．

企業における新製品開発のための研究などでは，製品の特性に影響する要因が技術的に絞り込めないことが多いので，実験で取り上げる因子数が多くなることは珍しくない．そこで，時間や費用の点から，もっと少ない回数で必要な情報が得られるような実験計画はできないか，という要請が出てくる．それに応えるのが直交表による実験である．

直交表には 2 水準系L_8，L_{16}，L_{24}，…と 3 水準系L_9，L_{27}，…があるが，今回は 3.1 節での設定に従い，2 水準系のL_8を用いる．これはすべて 2 水準系の因子について，総数 8 回の実験を行うものである．

総数 8 回の実験データの自由度 8-1=7 であるから，自由度 1 の 2 水準因子を 7 個まで取り組むことができるので，7 個の列で構成されている．直交表の内部には 1 と 2 の数字が 4 個ずつ並んでいるが，これが各実験での因子の水準を示す．

また直交表は，任意の 2 列に対して，それらの水準が等しい場合に 1，水準が異なる場合に 2 となった列が必ず含まれているから，その列から交互作用が得られる．すなわち，直交表においては，実験に用いる因子間の交互作用が特例の列で決まってしまうので，交互作用の求まる列に他の因子を割り付けてしまうと，両方が交絡してしまう．その結果，いずれの要因による効果なのかが分離できなくなることに注意して，実験の割付をしなければならない．

以上に留意して，表 3.1 の列(1，2，4)には主効果の因子を割り付け，列(3，5，7)には交互作用をそれぞれ割り付ける．全ての因子を割り付けた後の空いた所には，誤差因子 e を割り付ける．

表 3.1　直交表.

No. ＼ 列	1	2	3	4	5	6	7
1	1	1	1	1	1	1	1
2	1	1	1	1	1	1	1
3	1	1	1	2	2	2	2
4	1	1	1	2	2	2	2
5	1	2	2	1	1	2	2
6	1	2	2	1	1	2	2
7	1	2	2	2	2	1	1
8	1	2	2	2	2	1	1

次に因子の割り付け方法は下記のとおりである.

- 「CAB 適正検査の得点高低」を因子 A として, 高いを第 1 水準, 低いを第水準 2 とする.
- 「BGM の有り無し」を因子 B として, 有りを第 1 水準, なしを第 2 水準とした.
- 「コードレビュー時間の長短」を因子 C として, 15 分を第 1 水準, 25 分を第 2 水準とする.
- 3 因子間の交互作用を A×B, A×C, B×C として割り付ける.
- それ以外の誤差要因を因子 e とする.
- 実験で得られたデータを外側に割り付ける.

3.4 実験の採点方法

本研究では，実験で得られたエラー摘出個数を直交表に直接割り当てをしなかった．なぜなら，バグを取り出した数だけを摘出してしまうと，正しい所を間違った指摘をしてしまったケースを無視してしまうからである．従って，今回の採点条件は以下の様に設定した．

- 正しくプログラムを指摘できた．
  → 5 点
- 指摘箇所が間違っているが正しくプログラム動いた．
  → 0 点
- 指摘箇所は合っているが正しくプログラムが書けなかった．
  → 3 点
- 指摘箇所も間違っていて正しくプログラムが動かなかった為，新たなバグを生み出した．
  → －5 点

として，採点を行った．表 3.1 の直交表に割り当てた（表 3.2 を参照）．

これらのデータを基に，データ解析を行い，人的要因と交互作用を明らかにする．

表 3.2　データを割り付けた直交表.

No. \ 因子	CAB A	BGM B	A×B	レビュー時間 C	A×C	e	B×C	採点後の成績		
1	1	1	1	1	1	1	1	35	31	23
2	1	1	1	2	2	2	2	5	20	18
3	1	2	2	1	1	2	2	28	20	15
4	1	2	2	2	2	1	1	30	20	35
5	2	1	2	1	2	1	2	10	5	8
6	2	1	2	2	1	2	1	0	0	13
7	2	2	1	1	2	2	1	5	−5	−5
8	2	2	1	2	1	1	2	5	13	0

4. 分散分析によるデータ解析

4.1 分散分析と有意要因の推定

直交表と当てはめたデータから，分散分析を以下の手順で行う．

(1) 全変動を求める．

(2) 各列の変動を求める．

(3) 誤差変動を求める．

(4) 分散比を求め，F 検定により各因子の有意性を検討する．

(5) 純変動を求める．

(6) 寄与率を求める．

(7) 分散分析表を作成する．

(8) 最適条件を選定する．

(9) 有意要因の推定を行う．

以下の手順で分散分析をした結果を表 4.1 に示す．

表 4.1 分散分析．

	要因	平方和	自由度	分散	分散比	検定	P値
1	A	2223.4	1	2223.4	51.706	**	0
2	B	2.042	1	2.042	0.047		0.83
3	AB	63.375	1	63.375	1.474		0.242
4	C	5.042	1	5.042	0.117		0.736
5	AC	57.042	1	57.042	1.327		0.266
6	BC	425.04	1	425.04	9.885	**	0.006
7	ABC	51.042	1	51.042	1.187		0.292
8 誤差	e	688	16	43			
9	計	3515	23				

[検定結果] ＊＊:1%有意 ＊:5%有意 空白:有意差なし

このままでは，検定の感度が悪いため，プーリングをした結果を，表 4.2 にまとめる．

表 4.2　プーリング後の結果．

	要因	平方和	自由度	分散	分散比	検定	P値
1	A	2223.4	1	2223.4	49.152	**	0
2	B	2.042	1	2.042	0.045		0.834
3	C	5.042	1	5.042	0.111		0.742
4	BC	425.04	1	425.04	9.396	**	0.006
5 誤差	e	859.46	19	45.235			
6	計	3515	23				

[検定結果]　＊＊:1%有意　＊:5%有意　空白:有意差なし

結果として，要因 A と要因 B，要因 C との交互作用（BC）が高度に有意となった．このうち，後者についての要因効果の概略を図 4.1 と図 4.2 にそれぞれ示す．

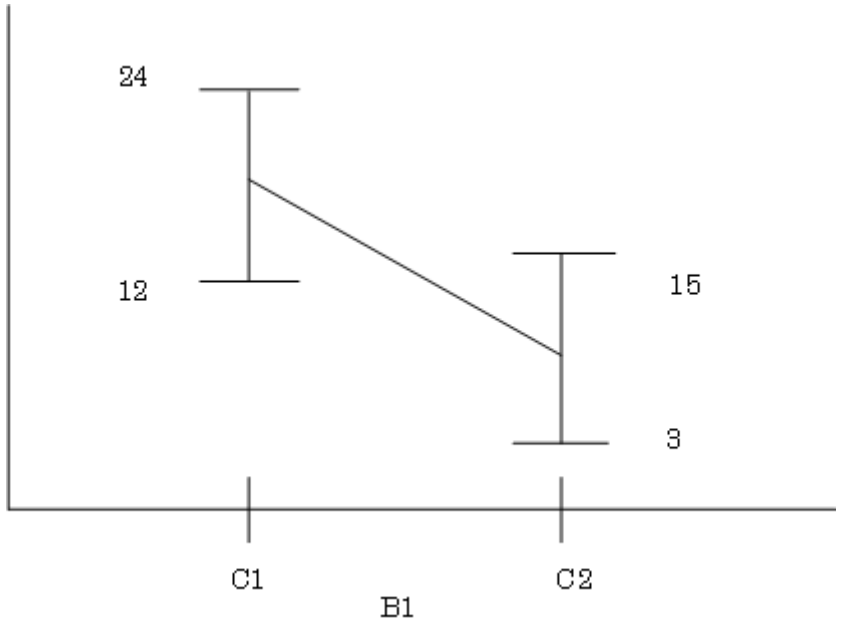


図 4.1　要因効果(音有り，時間の長 C1・短 C2)．

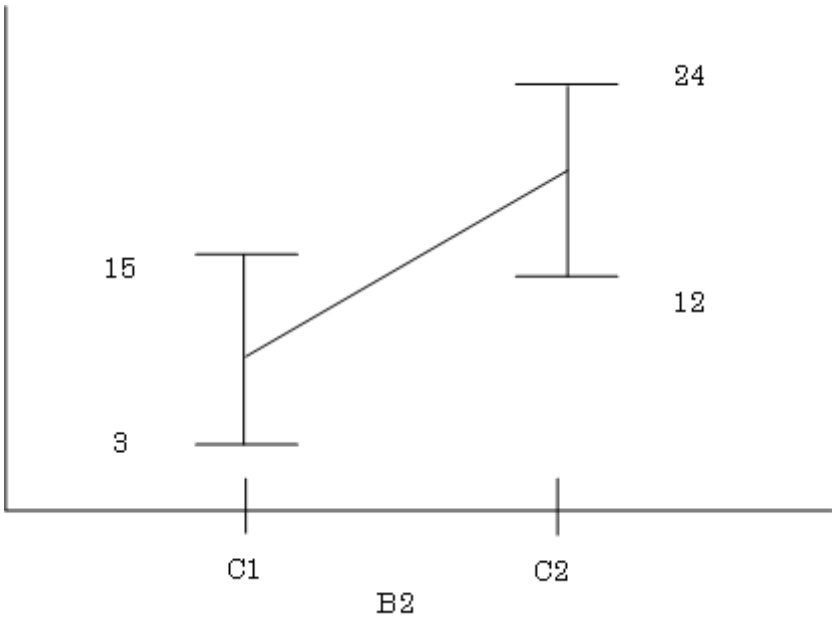


図 4.2 (音無し, 時間の長 C1・短 C2).

4.2 分析結果と考察

分散分析結果である表 4.1 を基に以下のことが確認できた.

(1) CAB 適正検査の得点高低は, フォールトの指摘に有意であった. このことから CAB の点数が高い人ほど, フォールトが発見でき, CAB の点数が低い人ほど, フォールトが発見できない.

(2) BGM の有り無しでは, フォールトの発見に有意ではなかった.

(3) 時間の長短では, フォールトの発見に有意ではなかった.

(4) 各因子の交互作用に有意性が見られた.

図 4.1 と図 4.2 から, CAB の点数が低く, BGM がある作業環境では, 時間が長い方がフォールトの発見個数が良かった. また, CAB の点数が低く, BGM が無い作業環境では時間が短い方が良いと検出された. これは, 作業が長くなることによる精神的な苦痛を BGM が緩和されることでフォールトの発見に有意になったのではないかと考えられる. これらの交互作用についての更なる

検討については，今後の課題とする．

実験を行う前では，CABの点数が高い人は，与えられた時間が長ければ長いほど，フォールトを発見個数が増えると考えられた．しかし，今回の実験ではCABと時間の長さに関係がなかった．問題のプログラムの難易度を変えていれば結果は変わっていたのではないかと考えられる．

5. 共分散分析によるデータ解析

5.1 共分散分析の概要[5]

共分散分析は分散分析と違い，分析の精度や偏りを除去するための変数を導入して分析をする．この変数のことを共変量という．この共変量を考慮に入れてデータ解析を行う．

データは変量と共変量２つある．変量には実験のデータを，共変量に CAB の点数を割り当てる． 回帰直線を利用して分散分析をさらに一般化した手法を共分散分析と呼ぶ．回帰直線は各水準によって決まるので，水準ごとの回帰を同時に分析することとなる．

ここでは共分散分析の概要と適用する際の条件を説明する．

(1)各水準の回帰直線

水準に対して，

$$y_{ij}=\alpha_i+\beta_i x_{ij}+\varepsilon_{ij} \tag{5.1}$$

という回帰モデルを仮定する．図 5.1 に概念図を示す．

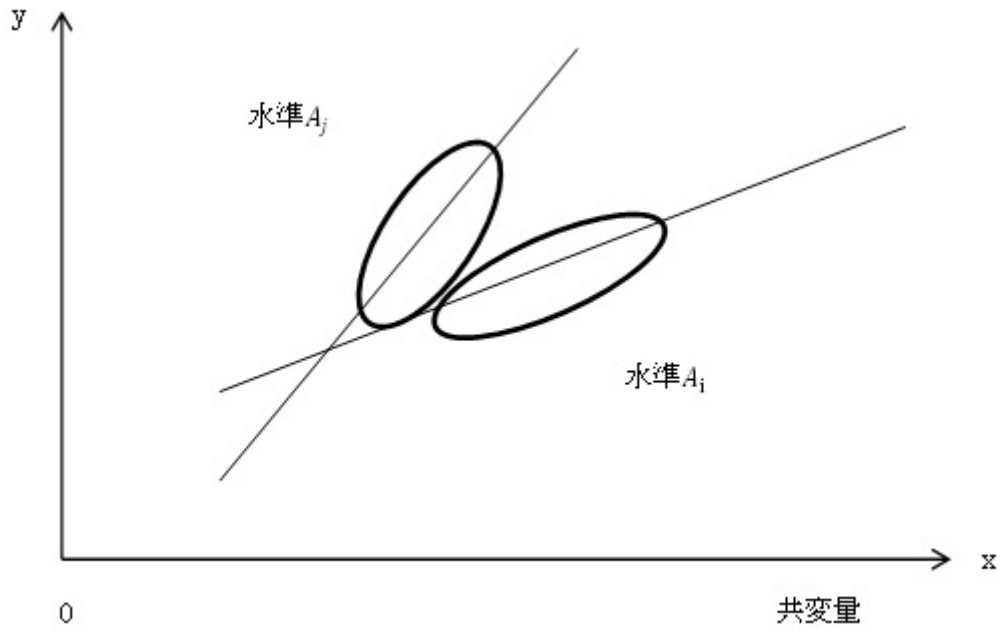


図 5.1 共分散分析の概要．

(2)回帰直線の平行性の検定

いずれの回帰直線もバラバラなので差の比較はできない．検定

をし，回帰直線の傾きが等しいと仮定しても良いことを確認する．図 5.2 はその概念図である．

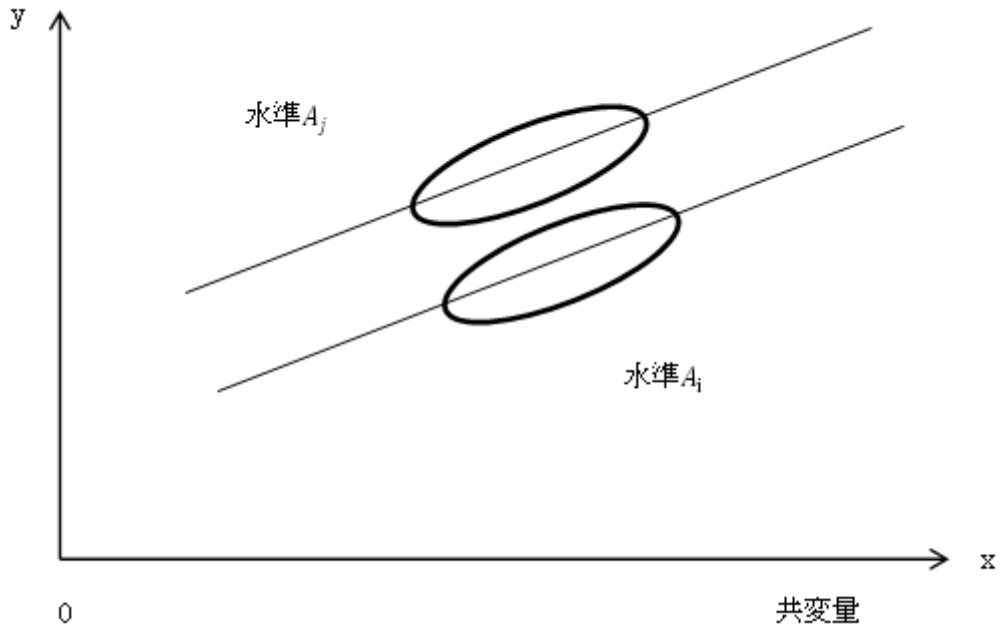


図 5.2　平行性の検定が棄却される場合．

(3)回帰の有意性の検定

回帰直線の傾きが等しいと仮定したのだが,傾きが 0 の場合は，通常の分散分析となり共変量を入れた意味がなくなってしまう．したがって傾きが 0 か否かの検定を行う．

(4)水準の差の検定

以上の検定が全て通るとき，共分散分析の目的である差の検定が行える．

5.2　共分散分析の検証手順

今回検証するデータは，CAB の点数とフォールト発見個数で解析を行った．表 5.1 にデータを示す．

表 5.1 データ．

A1	成績(y)	cab(x)	A2	成績(y)	cab(x)
時間(長)	7	59.10	時間(短)	1	55.30
	5	58.30		4	86.20
	4	66.00		3	60.10
	6	70.10		6	54.50
	5	55.10		4	60.15
	4	58.30		7	65.10
	3	47.68		0	35.09
	1	53.80		0	53.99
	1	50.27		2	47.77
	2	35.21		1	48.18
	1	40.68		2	51.16
	0	53.00		0	49.29

回帰直線の平行性の検定を行う前に，まずは統計量の計算を行う．Nはデータ数とする．

$$CT_y=\frac{\left(\sum_{i=1}^{a}\sum_{j=1}^{n_i}y_{ij}\right)^2}{N} \tag{5.2}$$

$$CT_x=\frac{\left(\sum_{i=1}^{a}\sum_{j=1}^{n_i}x_{ij}\right)^2}{N} \tag{5.3}$$

$$CT_{yx}=\frac{\left(\sum_{i=1}^{a}\sum_{j=1}^{n_i}y_{ij}\right)\left(\sum_{i=1}^{a}\sum_{j=1}^{n_i}x_{ij}\right)}{N} \tag{5.4}$$

次に，全変動を求める

$$S_{Ty}=\sum_{i=1}^{a}\sum_{j=1}^{n_i}{y_{ij}}^2-CT_y \tag{5.5}$$

$$S_{Tx}=\sum_{i=1}^{a}\sum_{j=1}^{n_i}{x_{ij}}^2-CT_x \tag{5.6}$$

$$S_{Txy}=\sum_{i=1}^{a}\sum_{j=1}^{n_i}y_{ij}\,x_{ij}-CT_{xy} \tag{5.7}$$

変動を求め，水準間変動と水準内変動を求める．

$$S_{Ay}=\frac{\left(\sum^{n_1}y_{1j}\right)^2}{n_1}+\frac{\left(\sum^{n_2}y_{2j}\right)^2}{n_2}+\cdots+\frac{\left(\sum^{n_a}y_{aj}\right)^2}{n_a}-CT_y \tag{5.8}$$

$$S_{Ax}=\frac{\left(\sum^{n_1}x_{1j}\right)^2}{n_1}+\frac{\left(\sum^{n_2}x_{2j}\right)^2}{n_2}+\cdots+\frac{\left(\sum^{n_a}x_{aj}\right)^2}{n_a}-CT_x \tag{5.9}$$

$$S_{Ayx}=\frac{\left(\sum^{n_1}y_{1j}\right)\left(\sum^{n_1}x_{1j}\right)}{n_1}+\frac{\left(\sum^{n_2}y_{2j}\right)\left(\sum^{n_2}x_{2j}\right)}{n_2}+\cdots+\frac{\left(\sum^{n_a}y_{aj}\right)\left(\sum^{n_a}x_{aj}\right)}{n_a}-CT_{yx} \tag{5.10}$$

$$S_{Ey}=S_{Ty}-S_{Ay} \tag{5.11}$$

$$S_{Ex}=S_{Tx}-S_{Ax} \tag{5.12}$$

$$S_{Eyx}=S_{Tyx}-S_{Ayx} \tag{5.13}$$

水準A_1については

$$\frac{\left(n_1\Sigma_{j=1}^{n_1}x_{1j}y_{1j}-\left(\Sigma_{j=1}^{n_1}y_{1j}\right)\left(\Sigma_{j=1}^{n_1}x_{1j}\right)\right)^2}{n_1\left(n_1\Sigma_{j=1}^{n_1}{x_{1j}}^2-\left(\Sigma_{j=1}^{n_1}x_{1j}\right)^2\right)} \tag{5.14}$$

であり，水準A_aについては

$$\frac{\left(n_a\Sigma_{j=1}^{n_a}x_{aj}y_{aj}-\left(\Sigma_{j=1}^{n_a}y_{aj}\right)\left(\Sigma_{j=1}^{n_a}x_{aj}\right)\right)^2}{n_a\left(n_a\Sigma_{j=1}^{n_a}{x}^2-\left(\Sigma_{j=1}^{n_a}x_{aj}\right)^2\right)} \tag{5.15}$$

$$S_{Bx}=A_1+A_2\cdots A_a \tag{5.16}$$

が成立する．

非平行性の変動について，以下に従って諸量を求める．

平方和

$$S_{NP}=S_{Bx}-\frac{\left(S_{Eyx}\right)^2}{S_{Ex}} \tag{5.17}$$

に対して自由度は$a-1$であるから，平均平方は

$$V_{NP}=\frac{S_{NP}}{a-1} \tag{5.18}$$

となる．

残差変動についても同様に，平方和は

$$S_{E'}=S_{Ex}-S_{Bx} \tag{5.19}$$

と与えられ，自由度は$N-2a$であるから，平均平方は

$$V_{E'}=\frac{S_{E'}}{N-2a} \tag{5.20}$$

となる．

これらにより，

仮説$H_0:\beta_{1=}\beta_{1=}\cdots=\beta_a$ (5.21)

に対し，

$$\frac{V_{NP}}{V_{E'}}\geq F_{(a-1,N-2a)} \tag{5.22}$$

が棄却されなければ，水準$A_1,A_2,\cdots A_a$における回帰直線が互いに平行になっていると仮定してよい．

次に回帰の有効性の検定の手順を説明する．

回帰による変動の平均平方は，

$$V_R = \frac{(S_{Eyx})^2}{S_{Ex}} \tag{5.23}$$

で与えられ，水準内変動の平均平方は，

$$V_E = \frac{S_{Ey}S_{Ex} - (S_{Eyx})^2}{(N-a-1)S_{Ex}} \tag{5.24}$$

となる．

仮説$H_0: \beta = 0$に対し，

$$\frac{V_R}{V_E} \geq F_{(1,N-a-1)} \tag{5.25}$$

となる．これが棄却されると，$\beta = 0$でないので回帰モデルとして採用して分析することができる．これから水準間の差を検定する．

$$S_A = \left(S_{Ty} - \frac{(S_{Tyx})^2}{S_{Tx}}\right) - \left(S_{Ey} - \frac{(S_{Eyx})^2}{S_{Ex}}\right) \tag{5.26}$$

より，水準間の変動の平均平方和は

$$V_A = \frac{S_A}{a-1} \tag{5.27}$$

となる．最終的に，仮説$H_0: a_1 = a_2 = \cdots = a_a$に対し，

$$\frac{V_A}{V_E} \geq F_{(1,N-a-1)} \tag{5.28}$$

が棄却されれば，水準$A_1, A_2, \cdots A_a$の間に差があるということになる．

5.3 分析の結果

表 5.1 のデータを用いて，時間を長・短の水準を 2 つに分けて分析してみたが，時間は有意となった．テスト時間が長いグループについて，共分散分析によって求まる回帰直線は，

$$\mathrm{y} = -3.3933 + 0.1231x \tag{5.29}$$

となった．ここでxは CAB の得点であり，yはレビューの成績である．この回帰式から，実際の実験参加者 12 名のデータ(CAB 得点，成績)との成績とyの値のずれwを抽出する．wが従う正規分布$N(\mu, \sigma^2)$を推定したところ，

$$\hat{\mu} = -7.34 \times 10^{-6} \approx 0 \tag{5.30}$$

$$\hat{\sigma} = 1.70164 \tag{5.31}$$

が得られた．従って概ねwは$N(0, 1.70164^2)$に従っていると考えて良い．

このことから仮想的なレビュー担当者を仮定し，その CAB の得点xが得られたものとすると，この実験環境下では，このレビュー得点の分布は

$$N(-3.3933 + 0.1231 \times x, 1.70164^2) \tag{5.32}$$

に従うと考えることができる．これは CAB の得点とレビュー成績を結びつけるという意味において有用な情報となりうる．

同様に，テスト時間が短いグループに対して分析した結果，wが従う分布$N(\mu, \sigma^2)$は，

$$\hat{\mu} = -9.325 \times 10^{-6} \approx 0 \tag{5.33}$$

$$\hat{\sigma} = 1.85521 \tag{5.34}$$

となった．これよりレビュー時間が短いことによるレビュー成績の CAB による分布は

$$N(-4.34231 + 0.1231 \times x, 1.85521^2) \tag{5.35}$$

により表現できることがわかる．

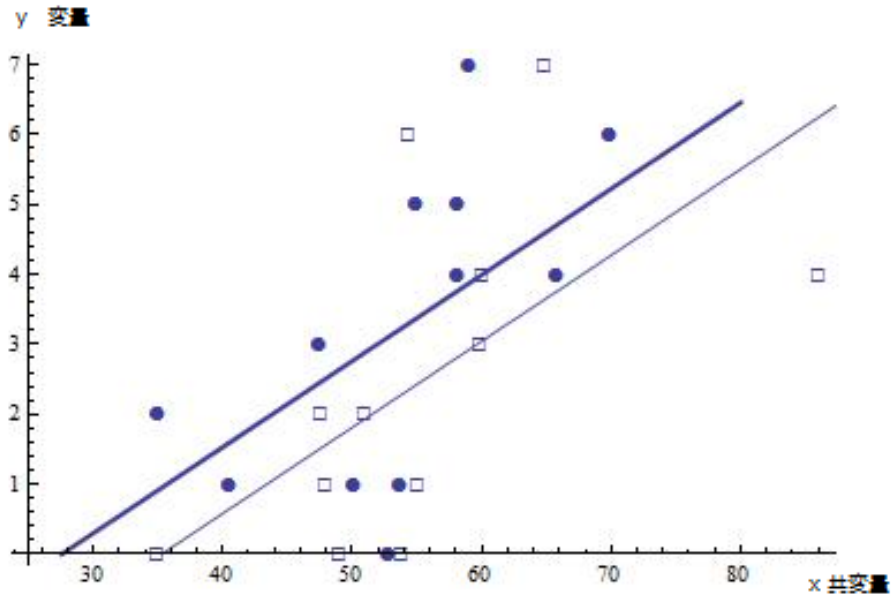


図 5.3 共分散分析の結果（●：水準 1，□：水準 2）．

6. おわりに

本研究から，分散分析ではCABの有意性を見ることができた．これは，予想していた結果とあっていた．単体での有意はCAB以外でなかったが，時間と音の交互作用が見られた．また，共分散分析により，CABを共変量においた所，時間も有意となった．このことからCABは実験の成績に関わっていることがわかる．また，時間の有意が出た理由としては，実験での時間を長くすると，発見個数も多くなるが，訂正しなくても良い場所を訂正しまっていることがわかる．これにより，時間が長くなることにより，新たなバグを作ってしまうので，作業時間の設定が非常に大切であることがわかる．また，ソースコードの量によりコードレビューの時間は変わっていくのは容易に推測できる．なので，プログラムの量による最適なコードレビューの時間を，定量的に解析をして求める必要性があるのではないかと考えられる．

参考文献

[1] 山田茂，影山高章，木村光宏，高橋宗雄：「コードレビューにおける人的エラーと人的要因に関する考察」，電子情報通信学会論文誌，信学論(A), Vol. J81-A, No.9, pp. 1238-1246，1998.
[2] 江崎和博，山田茂，高橋宗雄：「設計レビューにおけるソフトウェア信頼性に影響を及ぼす人的要因の品質工学的解析」，電子情報通信学会論文誌，信学論(A), Vol. J84-, No. 2, pp. 218-228, 2001.
[3] 日原圭祐：「品質工学的アプローチに基づくソフトウェア信頼性に影響を及ぼす人的要因に関する研究」，鳥取大学院工学研究科修士論文，2000.
[4] SPI ノート会：CAB・GAB 完全突破法!(2014 年度版)，洋泉社，2014.
[5] 石村貞夫：分散分析のはなし，東京図書館，1992.

謝辞

本研究をすすめるにあたり多大なご協力を頂いた本学院工学研究科システム工学専攻，木村光宏教授に感謝致します．

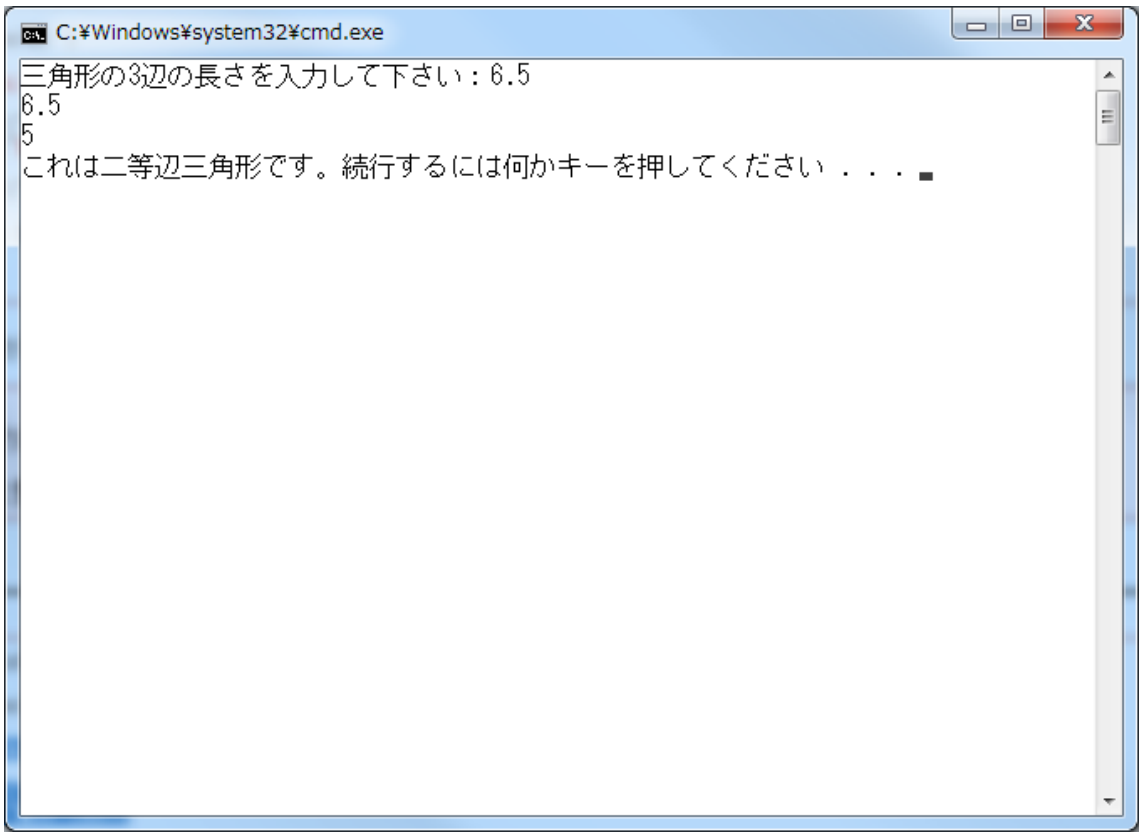
付録

実験に用いた２つの問題プログラムと仕様書をそれぞれ簡単に付す．

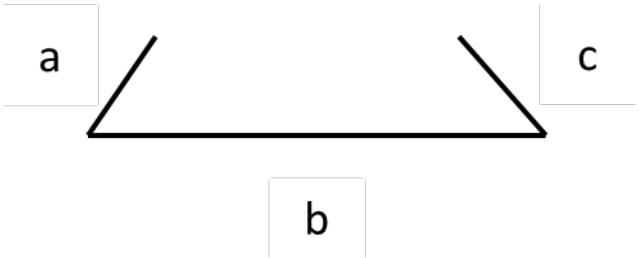
仕様書

三角形の判別プログラムです．
三角形の三辺を入力することで，正三角形，二等辺三角形，直角三角形，直角二等辺三角形，三角形ではない内のどんな三角形かを判別し，画面に結果を出力します．

実行画面
二等辺三角形の場合



このうち「三角形ではない」については，例えば



の様な場合です．

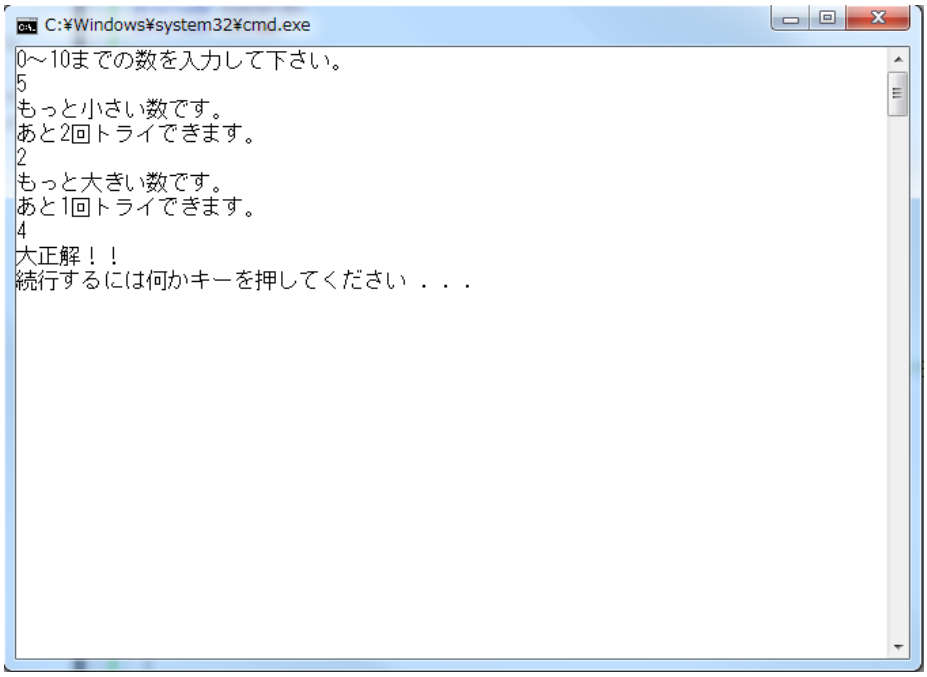
```
#include <stdio.h>
#include <cstdlib>
#include <ctime>

 int main(){
 srand((unsigned)time(NULL));
 int s = rand()%10;
 int temp;
 printf("0～10までの数を入力して下さい。¥n");

 for(int i=0; i<2; i++){
 scanf("%d", temp);
 if(temp = 0 && temp <= 10){
 if(temp = s){
 printf("大正解！！¥n");
 goto END;
 }
 if(temp > s){
 printf("もっと小さい数です。");
 printf("あと%d回トライできます。¥n", 2-i);
 }else{
 printf("もっと大きい数です。");
 printf("あと%d回トライできます。¥n", 2-i);
 }
 }
 else{
 printf("入力エラー¥n");
 return 0;
 }
 }
 printf("残念でした…¥n");
 END:
 return 0;
 }
```

仕様書

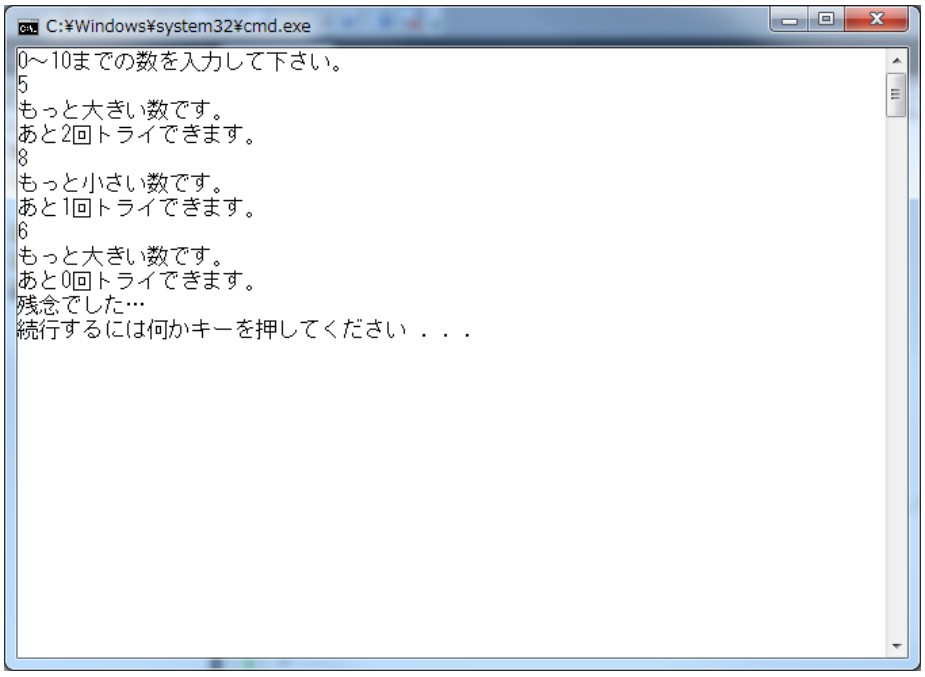
数字当てゲームのプログラムです．
0～10までの数のうち，任意の数字を3回まで入力して正解の数を当てるものです．

実行画面

正解の場合



不正解の場合



```
#include <stdio.h>
#include <cstdlib>
#include <ctime>

int main(){
 srand((unsigned)time(NULL));
 int s = rand()%10;
 int temp;
 printf("0～10までの数を入力して下さい。¥n");

 for(int i=0; i<2; i++){
 scanf("%d", temp);
 if(temp = 0 && temp <= 10){
 if(temp = s){
 printf("大正解！！¥n");
 goto END;
 }
 if(temp > s){
 printf("もっと小さい数です。");
 printf("あと%d回トライできます。¥n", 2-i);
 }else{
 printf("もっと大きい数です。");
 printf("あと%d回トライできます。¥n", 2-i);
 }
 }
 else {
 printf("入力エラー¥n");
 return 0;
 }
 }
 printf("残念でした…¥n");
 END:
 return 0;
 }
```