

Ужин с ПК. Математика в тестировании

Андрей Ершов, Виталий Брагилевский, Дарья Манухина, Иван Пономарёв

Нужна ли математика в тестировании?

«Стать тестировщиком за три дня»

vc.ru › 87842-s-chego-nachat-testiro... ▾ [Translate this page](#)

С чего начать тестировщику: материалы для старта - VC.ru

Oct 14, 2019 — Я не думал особо долго на счет того, с чего начать мой ... Вернитесь к книгам; Осознайте, что учиться надо каждый **день** ... Я к тому, что можно **стать** хорошим **тестировщиком** без знаний английского вообще.

software-testing.ru › edu › schedule ▾ [Translate this page](#)

Онлайн-интенсив для начинающих тестировщиков (3-х ...

Хотите **стать тестировщиком**? ... 2 лекции в неделю — около часа в **день**. ... Тестировщиком за неделю (или **три**) вы не станете, только изучите основы ...

smartprogress.do › goal ▾ [Translate this page](#)

Тестировщик за 3 дня - SmartProgress

Jun 27, 2016 — Я думаю, что этих ресурсов мне хватит **на 3 дня**, чтобы **стать** более начитанной по тестированию. Подтягивать планирую только ...

| | |
|------------------------------|------|
| Регион | |
| Россия | 1179 |
| Москва | 486 |
| Санкт-Петербург | 192 |
| Еще 82 | |
| Уровень дохода | |
| Указан | 441 |
| от 65 000 руб. | 317 |
| от 130 000 руб. | 163 |
| от 190 000 руб. | 95 |
| от 250 000 руб. | 54 |
| от 315 000 руб. | 20 |
| Профобласть | |
| IT, телеком | ✕ |
| Специализация | |
| Программирование, Разработка | 708 |
| Аналитик | 448 |
| Инженер | 227 |
| Еще 34 | |
| Отрасль компании | |
| Опыт работы | |
| Тип занятости | |

Будьте первыми

Математик-аналитик (data scientist / data analyst)

от 85 000 руб.

ООО ННФормат ✓

Санкт-Петербург

Анализ и прогнозирование временных рядов. Разработка и построение статистических и ML-моделей. Feature Engineering. Формирование продуктовых метрик под бизнес-задачи.

Высшее профильное образование (бакалавриат, специалитет или магистратура) в области прикладной **математики** и информатики. Знание основных статистических моделей и методов машинного...

[Откликнуться](#)

2 ноября

Будьте первыми

Математик / Программист

120 000-180 000 руб.

ООО Специальный Технологический Центр ✓

Санкт-Петербург, ● Площадь Мужества

Изучение и разработка алгоритмов из области 3D реконструкции и картографии. Реализация и оптимизация алгоритмов на языке C++.

...3D графика, картография, 3D реконструкция). Желательно - опыт использования технологий CUDA/OpenCL. Образование: высшее техническое / **математика** / физика / прикладная **математика**.

[Откликнуться](#)

2 ноября

Отклик без резюме

Будьте первыми

Математик-программист

АО Институт Оргэнергострой ✓

Многие общематематические концепции переходят в практику

- «Техника анализа классов эквивалентности»

$$a \sim a$$

$$a \sim b \Rightarrow b \sim a$$

$$a \sim b \wedge b \sim c \Rightarrow a \sim c$$

$$[a] = \{x \in X \mid x \sim a\}$$

Комбинаторика

- **Дано:** чистая функция с тремя булевыми параметрами.
- **Вопрос:** Сколько тест-кейсов надо написать, чтобы полностью протестировать эту функцию?

Комбинаторика

- **Дано:** чистая функция с тремя булевыми параметрами.
- **Вопрос:** Сколько тест-кейсов надо написать, чтобы полностью протестировать эту функцию?

$$2^3 = 8$$

Комбинаторика

- **Дано:** чистая функция с параметром в виде строки в алфавите a длиной не больше m ?
- **Вопрос:** Сколько тест-кейсов надо написать, чтобы полностью протестировать эту функцию?

Комбинаторика

- **Дано:** чистая функция с параметром в виде строки в алфавите a длиной не больше m ?
- **Вопрос:** Сколько тест-кейсов надо написать, чтобы полностью протестировать эту функцию?

$$1 + a + a^2 + \dots + a^m = \frac{a^{m+1} - 1}{a - 1}$$

Комбинаторика

- **Дано:** три браузера, три ОС, три сценария.
- **Вопрос:** так сколько нам нужно тестов? и каким количеством тестов мы готовы обойтись?

Комбинаторика

- **Дано:** три браузера, три ОС, три сценария.
- **Вопрос:** так сколько нам нужно тестов? и каким количеством тестов мы готовы обойтись?

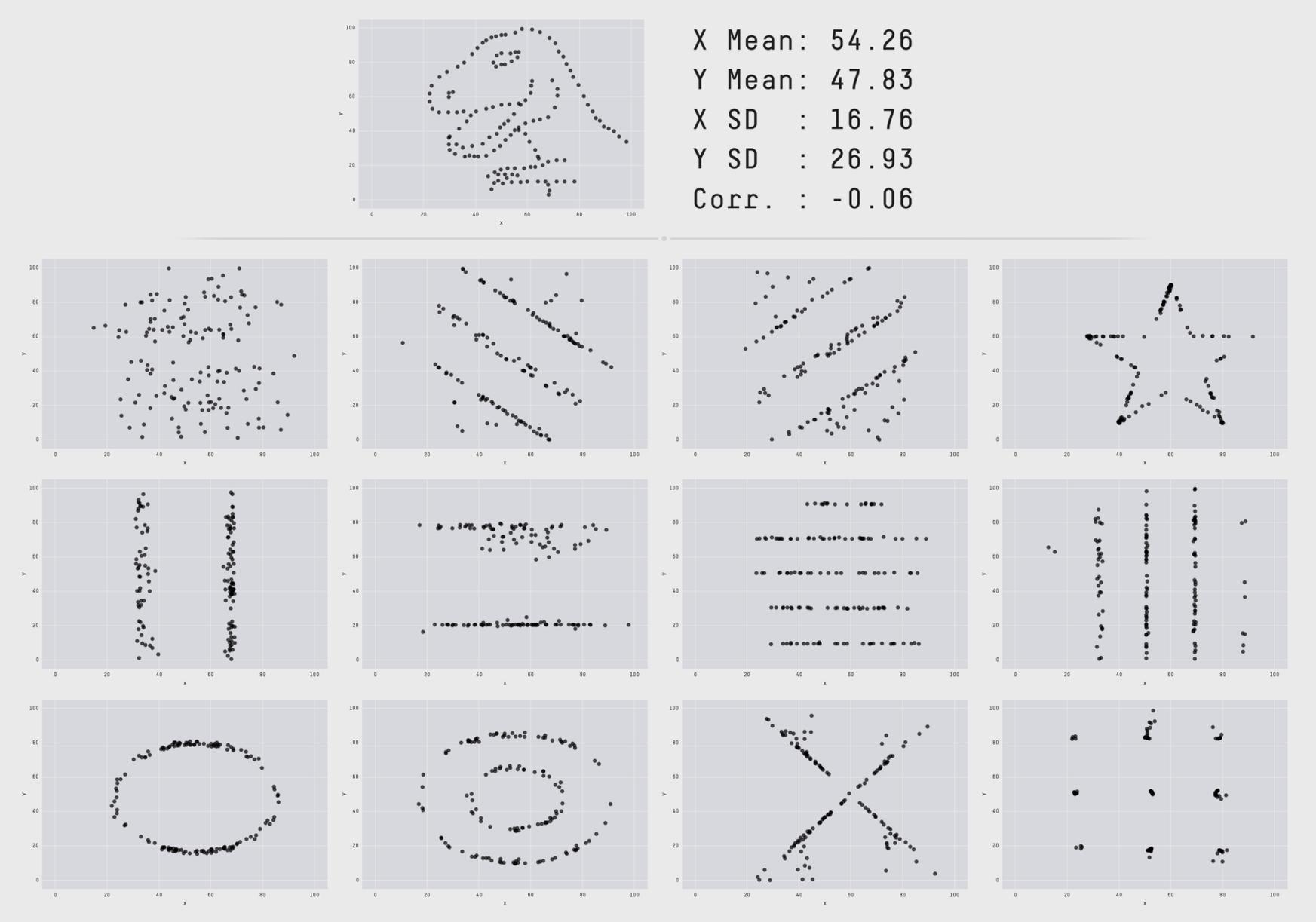
| | | |
|---|---|---|
| А | В | С |
| С | А | В |
| В | С | А |

Что почитать-посмотреть

- **Lee Copeland.** [A Practitioner's Guide to Software Test Design](#)
- **Мария Палагина.** [Жизнь МИКРОбов: Heisenbug-Piter 2020](#)
- Всё о pairwise testing: <http://pairwise.org>

Статистика

- **Дано:** Из мониторинга мы взяли ~ 1 млн замеров latency за период A и ещё ~ 1 млн замеров latency за период B .
- **Вопрос:** Как понять, стала ли система работать лучше/хуже/ничего не изменилось?



1. Datasaurus Dozen, <https://www.autodesk.com/research/publications/same-stats-different-graphs>

Что посмотреть?

Узнаем больше про статистику с Анатолием Карповым:

<https://stepik.org/course/76/promo>

The screenshot shows the Stepik website interface for the course 'Основы статистики'. At the top, there is a navigation bar with the Stepik logo, 'Каталог', 'Преподавание', a search bar, and language options ('Русский', 'Войти', 'Регистрация'). The main content area features the course title 'Основы статистики' and a detailed description: 'Курс знакомит слушателей с основными понятиями и методами математической статистики. В течение трех недель мы рассмотрим наиболее широко используемые статистические методы и принципы, стоящие за ними. Полученных знаний будет достаточно для решения широкого круга задач, возникающих в рамках исследовательской работы.' To the right of the text is a video player with a play button and the text 'Начать просмотр'. Below the description, there are icons for '3-4 часа в неделю' and 'Сертификат Stepik'. At the bottom right of the course card, there is a rating of 4.9 stars, '1 879 отзывов', and '132 719 учащихся'.



О курсе

В рамках трехнедельного курса рассматриваются подходы к описанию получаемых в исследованиях данных, основные методы и принципы

Бесплатно

Поступить на курс

начало 5 ноября 2020 г.

Доклады (для тех, кто уже разбирается в статистике)

- **Андрей Акиншин.** Heisenbug-Piter 2020. [Анализируем перформанс с пользой для себя и окружающих](#)
- **Андрей Акиншин, Андрей Паньгин, Алексей Шипилёв** Techtrain: Вторая волна. [О перформансе серьезно.](#)
- **Jack Vanlightly.** [System testing of RabbitMQ](#)

Логика

Что это такое?

$$\forall n \in \mathbb{N} (L(n) \iff (n:16) \vee (n:4 \wedge \neg n:25))$$

Логика

Что это такое?

$$\forall n \in \mathbb{N} (L(n) \iff (n:16) \vee (n:4 \wedge \neg n:25))$$

Является ли n номером високосного года в Григорианском календаре

Что посмотреть?

- **Kevlin Henney**. Heisenbug-Moscow 2020. [What we talk about when we talk about unit testing](#)
- **Jessica Ingrassellino**. Heisenbug-SPb 2019. [Property testing: Strategic automation for devs and SDETs](#)



$$\forall p \in P \diamond \neg \text{hungry}(p)$$

АРТ. БЕЛЕВИЧ

Алгоритм dining_philosophers

```
algorithm dining_philosophers
  variables chopsticks = [chopstick \in 1..NP |-> NULL]
  define
    LeftChopstick(p) == p
    RightChopstick(p) == IF p = NP THEN 1 ELSE p + 1
    HeldChopsticks(p) == { x \in {LeftChopstick(p), RightChopstick(p)}: chopsticks[x] = p}
    AvailableChopsticks(p) == { x \in {LeftChopstick(p), RightChopstick(p)}: chopsticks[x] = NULL}
  end define;
  process philosopher \in 1..NP
    variables hungry = TRUE;
    begin P:
      while hungry do
        with chopstick \in AvailableChopsticks(self) do
          chopsticks[chopstick] := self;
        end with;
        Eat:
        if Cardinality(HeldChopsticks(self)) = 2 then
          hungry := FALSE;
          chopsticks[LeftChopstick(self)] := NULL ||
          chopsticks[RightChopstick(self)] := NULL;
        end if;
      end while;
    end process;
  end algorithm;
```

TLA+

$\{x \in \{LeftChopstick(p), RightChopstick(p)\} : chopsticks[x] = NULL\}$

VARIABLE *hungry*

vars \triangleq $\langle chopsticks, pc, hungry \rangle$

ProcSet \triangleq $(1 .. NP)$

Init \triangleq
Global variables
 $\wedge chopsticks = [chopstick \in 1 .. NP \mapsto NULL]$
Process philosopher
 $\wedge hungry = [self \in 1 .. NP \mapsto TRUE]$
 $\wedge pc = [self \in ProcSet \mapsto "P"]$

P(self) \triangleq $\wedge pc[self] = "P"$
 \wedge IF *hungry*[*self*]
 THEN $\wedge \exists chopstick \in AvailableChopsticks(self) :$
 $chopsticks' = [chopsticks \text{ EXCEPT } ![chopstick] = self]$
 $\wedge pc' = [pc \text{ EXCEPT } ![self] = "Eat"]$
 ELSE $\wedge pc' = [pc \text{ EXCEPT } ![self] = "Done"]$
 $\wedge chopsticks' = chopsticks$
 \wedge UNCHANGED *hungry*

Model Checking

Deadlock ?

Deadlock reached.

Error-Trace Exploration +

Error-Trace 🔍 📄 🔄

| Name | Value |
|--|--|
| <ul style="list-style-type: none"> ▶ forks ▶ hungry ▶ pc | <<NULL, NULL, NULL, NULL>> <<TRUE, TRUE, TRUE, TRUE>> <<"P", "P", "P", "P">> |
| <ul style="list-style-type: none"> ▼ ▲ <P line 63, col 12 to line 70, col 30 of module Philosophers> ▶ forks ▶ hungry ▶ pc | State (num = 2) <<NULL, 2, NULL, NULL>> <<TRUE, TRUE, TRUE, TRUE>> <<"P", "Eat", "P", "P">> |
| <ul style="list-style-type: none"> ▼ ▲ <P line 63, col 12 to line 70, col 30 of module Philosophers> ▶ forks ▶ hungry ▶ pc | State (num = 3) <<1, 2, NULL, NULL>> <<TRUE, TRUE, TRUE, TRUE>> <<"Eat", "Eat", "P", "P">> |
| <ul style="list-style-type: none"> ▼ ▲ <Eat line 72, col 14 to line 79, col 47 of module Philosophers> ▶ forks ▶ hungry ▶ pc | State (num = 4) <<1, 2, NULL, NULL>> <<TRUE, TRUE, TRUE, TRUE>> <<"P", "Eat", "P", "P">> |
| <ul style="list-style-type: none"> ▼ ▲ <Eat line 72, col 14 to line 79, col 47 of module Philosophers> ▶ forks ▶ hungry ▶ pc | State (num = 5) <<1, 2, NULL, NULL>> <<TRUE, TRUE, TRUE, TRUE>> <<"P", "P", "P", "P">> |
| <ul style="list-style-type: none"> ▼ ▲ <P line 63, col 12 to line 70, col 30 of module Philosophers> ▶ forks ▶ hungry ▶ pc | State (num = 6) <<1, 2, 3, NULL>> <<TRUE, TRUE, TRUE, TRUE>> <<"P", "P", "Eat", "P">> |
| <ul style="list-style-type: none"> ▼ ▲ <P line 63, col 12 to line 70, col 30 of module Philosophers> ▶ forks ▶ hungry ▶ pc | State (num = 7) <<1, 2, 3, 4>> <<TRUE, TRUE, TRUE, TRUE>> <<"P", "P", "Eat", "Eat">> |
| <ul style="list-style-type: none"> ▼ ▲ <Eat line 72, col 14 to line 79, col 47 of module Philosophers> ▶ forks ▶ hungry ▶ pc | State (num = 8) <<1, 2, 3, 4>> <<TRUE, TRUE, TRUE, TRUE>> <<"P", "P", "Eat", "P">> |
| <ul style="list-style-type: none"> ▼ ▲ <Eat line 72, col 14 to line 79, col 47 of module Philosophers> ▶ forks ▶ hungry ▶ pc | State (num = 9) <<1, 2, 3, 4>> <<TRUE, TRUE, TRUE, TRUE>> <<"P", "P", "P", "P">> |

Нужны ли тесты, если есть формальная спецификация?

1. model checking vs. формальное доказательство (Isabelle и Coq).
2. synthesis vs. refinement
3. [Случай Microsoft Iron Fleet](#)
4. **D. Knuth**, letter to Dr. P. van Emde Boas, April 1977:

```
else if x > greatest[l] then greatest[l] ← x;  
end;
```

The implementation of deletion would be similar. It is safe to use 0 and $2^{16}-1$ for $-\infty$ and $+\infty$.

Beware of bugs in the above code; I have only proved it correct, not tried it.

Что почитать/посмотреть?

1. **Leslie Lamport.** Specifying Systems: The TLA+ Language
2. **Hillel Wayne.** Practical TLA+
3. **Jack Vanlightly.** Heisenbug-Piter 2019. A systematic approach to building reliable distributed systems

Так нужна ли математика в тестировании?
