

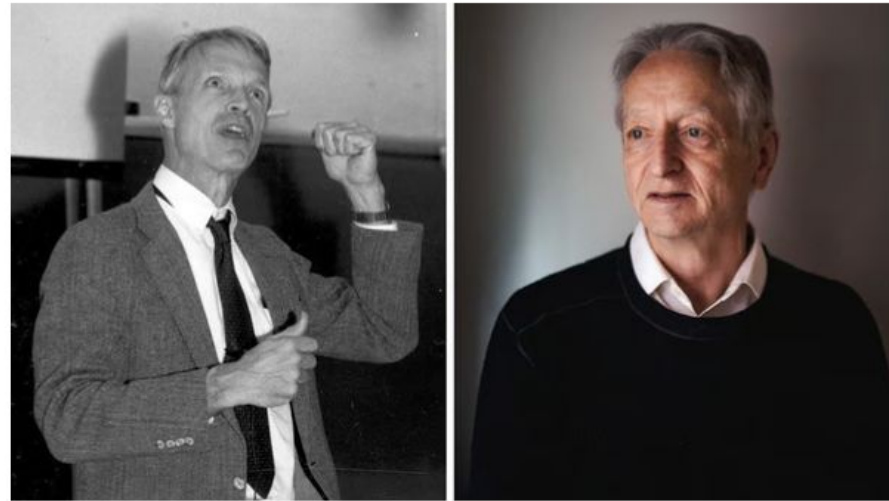
genAI@Dev

Wolfgang Pree

Science & technology | The 2024 Nobel prizes

# AI researchers receive the Nobel prize for physics

The award, to Geoffrey Hinton and John Hopfield, stretches the definition of the field



PHOTOGRAPH: AIP EMILIO SEGRÈ VISUAL ARCHIVE

Science & technology | The 2024 Nobel prizes

# Google's DeepMind researchers among recipients of Nobel prize for chemistry

The award honours protein design and the use of AI for protein-structure prediction



PHOTOGRAPH: IAN C. HAYDON/UW MEDICINE/GOOGLE DEEPMIND

# wie alles begann ...

- Debatte in den 1980er und 1990er Jahren:
  - Expertensysteme (Wenn-Dann-Regeln) versus
  - künstliche neuronale Netze (KNN)
- Geoffrey Hinton (ACM Turing Award + Nobelpreis) hatte bereits in den 1980er Jahren das Grundkonzept für Large Language Models (LLMs) erfunden und implementiert
  - [Will digital intelligence replace biological intelligence](#) (Oxford Univ., Feb. 2024)

# wie alles begann ...

- Debatte in den 1980er und 1990er Jahren:
  - Expertensysteme (Wenn-Dann-Regeln) versus
  - künstliche neuronale Netze (KNN)
- Geoffrey Hinton (ACM Turing Award + Nobelpreis) hatte bereits in den 1980er Jahren das Grundkonzept für Large Language Models (LLMs) erfunden und implementiert
  - [Will digital intelligence replace biological intelligence](#) (Oxford Univ., Feb. 2024)
- Er ist überzeugt, dass heutige LLMs Zusammenhänge tatsächlich die Welt verstehen und kreativ sind
- *generative AI (genAI)* hat sich als prägnantes Kürzel etabliert, was LLMs tun, Text, Bilder, Programme, Videos auf Basis eines sogenannten Prompt-Inputs erzeugen

# Typische GenAI Tools für Software-Entwicklung

- [GitHub Copilot](#)
  - seit 2021 verfügbar
  - PlugIn für die gängigen Programmierumgebungen  
=> kein Kontextwechsel, man bleibt 'in the zone'
  - signifikante Verbesserungen, insbesondere seit der Bereitstellung von ChatGPT 3 Ende 2022
- [OpenAI's ChatGPT](#) (4/4o/4o with canvas/o1-preview/o1-mini)
- [Google's Gemini](#)

Beispiel:

nicht dokumentierten Code verstehen und  
eine Einschränkung korrigieren

# Was macht folgender (Pseudo-)Code?

```
boolean forward; char x; char array c[1:20]; integer l, k, inc, term;
forward:= true; x:= RNC;
repeat l:= 0;
    repeat l:= l + 1; c[l]:= x; x:= RNC until x = sp ∨ x = pnt;
    while x = sp do x:= RNC;
    if forward then begin k:= 0; inc:= +1; term:= l end
        else begin k:= l + 1; inc:= -1; term:= l end;
    repeat k:= k + inc; PC(c[k]) until k = term;
    if x = pnt then PC(pnt) else PC(sp);
    forward:= ¬ forward
until x = pnt
```

RNC: char      read next character  
PC(ch: char)    print character

Konstante:

sp	<=> ' '	space
pnt	<=> '!	Punkt

# ChatGPT 4o with canvas

- korrekte Beschreibung, was der Code tut?
- Generierung eines Java-Programms, das dasselbe tut, aber besser lesbar ist
- Verwendung von Rekursion
- Generierung von Testfällen dazu



# GitHub Copilot

- Eingabe der Beschreibung, die wir erhalten hatten, als Kommentar
- Generierung von Testfällen

Beispiel:

Code aus natürlichsprachlicher Beschreibung  
generieren

## Advent of Code 2023, Tag 3, Gear Ratios

You and the Elf eventually reach a gondola lift station; he says the gondola lift will take you up to the water source, but this is as far as he can bring you. You go inside. It doesn't take long to find the gondolas, but there seems to be a problem: they're not moving. "Aaah!"

You turn around to see a slightly-greasy Elf with a wrench and a look of surprise. "Sorry, I wasn't expecting anyone! The gondola lift isn't working right now; it'll still be a while before I can fix it." You offer to help. The engineer explains that an engine part seems to be missing from the engine, but nobody can figure out which one. If you can add up all the part numbers in the engine schematic, it should be easy to work out which part is missing.

The engine schematic (your puzzle input) consists of a visual representation of the engine. There are lots of numbers and symbols you don't really understand, but apparently any number adjacent to a symbol, even diagonally, is a "part number" and should be included in your sum. (Periods (.) do not count as a symbol.)

Here is an example engine schematic:

```
467..114..  
...*.....  
..35..633..  
.....#...  
617*.....  
.....+.58..  
..592.....  
.....755..  
...$.*....  
.664.598..
```

In this schematic, two numbers are not part numbers because they are not adjacent to a symbol: 114 (top right) and 58 (middle right). Every other number is adjacent to a symbol and so is a part number; their sum is 4361.

Of course, the actual engine schematic is much larger. What is the sum of all of the part numbers in the engine schematic?

# zu schwierig?

- ChatGPT 4/4o
- Copilot
- ChatGPT 4o with canvas

Bachelorarbeit

*GenAI Applied to Selected  
Programming Tasks*

*An Evaluation Based on Case Studies*

Florian Frauenschuh  
Februar – August 2024

# Ergebnisse zur Advent-of-Code-Evaluierung 1, 2 und 3

Korrektheit des erzeugten Codes:

	ChatGPT-4	GitHub Copilot	Gemini Advanced
Day 1	✓	✓	✗
Day 2	✓	✓	✓
Day 3	✗	✗	✗

# Ergebnisse zur Advent-of-Code-Evaluierung 1, 2 und 3

## Qualitative Beurteilung des erzeugten Codes:

To summarize the quality of the code without factoring in correctness, the models are rated on a scale from 1 to 5, with 1 being the best and 5 being the worst grade.

- **ChatGPT-4:** Grade 1  
GPT's code introduces high modularity and is generally easy to read. The code is also descriptive, with the only major downside being some redundant comments.
- **GitHub Copilot:** Grade 4  
Often lacking understandability, due to a more monolithic approach and general non-descriptive code.
- **Gemini Advanced:** Grade 2  
Gemini's code quality is closer to GPT, but sometimes it could be a bit more descriptive.

# Copilot, Day 2: code snippet

```
BufferedReader br = new BufferedReader(new FileReader(file));

String st;
int sum = 0;
while ((st = br.readLine()) != null) {
    String [] parts = st.split(": ");
    int gameId = Integer.parseInt(parts[0].split(" ")[1]);
    String [] sets = parts[1].split("; ");
    boolean isPossible = true;
    for (String set : sets) {
        String [] cubes = set.split(", ");
        for (String cube : cubes) {
            String [] cubeParts = cube.split(" ");
            int count = Integer.parseInt(cubeParts[0]);
            String color = cubeParts[1];
            if ((color.equals("red") && count > 12) || (color.equals("green") && count > 13) || (color.equals("blue") && count > 14)) {
                isPossible = false;
                break;
            }
        }
        if (!isPossible) {
            break;
        }
    }
    if (isPossible) {
        sum += gameId;
    }
}
System.out.println(sum);
```



# GPT-4, Day 2: code snippet I

```
public static void main(String[] args) {
    String filePath = "./adventOfCode23/src/day2/input.txt";
    try {
        Scanner scanner = new Scanner(new File(filePath));
        int sumOfGameIds = 0;

        while (scanner.hasNextLine()) {
            String line = scanner.nextLine();
            if (isGamePossible(line)) {
                int gameId = Integer.parseInt(line.split(":")[0].split(" ")
                    [1]);
                sumOfGameIds += gameId;
            }
        }
        ...
    }
    ...
}
```

## GPT-4, Day 2: code snippet II

```
private static boolean isGamePossible(String gameData) {
    // Initial cube counts in the bag
    HashMap<String, Integer> bag = new HashMap<>();
    bag.put("red", 12);
    bag.put("green", 13);
    bag.put("blue", 14);

    String[] parts = gameData.split(":")[1].trim().split(";");
    for (String part : parts) {
        String[] cubes = part.trim().split(",");
        HashMap<String, Integer> counts = new HashMap<>();

        for (String cube : cubes) {
            String[] details = cube.trim().split(" ");
            int count = Integer.parseInt(details[0]);
            String color = details[1];

            // Update the temporary counts for this round
            counts.put(color, counts.getOrDefault(color, 0) + count);
        }

        // Check if the game round is possible
        for (String color : counts.keySet()) {
            if (counts.get(color) > bag.get(color)) {
                return false;
            }
        }
    }

    return true;
}
```

einfache Dart/Flutter App mit Login

# Prompts

Provide the complete source code for a ready-to-use Flutter application that allows users to create, edit and delete notes without using an external database service.

Flutter Version: 3.19.5

Provide the complete source code for a ready-to-use email and password authentication login flow in a Flutter application using Firebase.

Flutter Version: 3.19.6

# Ergebnisse

Notes app: ca. 150 LoC

- nur Copilot generiert korrekte und vollständige Lösung, Code ist jedoch monolithisch strukturiert
- Gemini hat das beste UI/UX generiert (Platzierung der Buttons, einfacher zum Editieren der Notizen)

Login: ca. 70 LoC

- Gemini liefert den am besten strukturierten Code, inkludiert 'Sign Up' Funktionalität, Code musste aber ergänzt/korrigiert werden
- Copilot generierte 'deprecated code'

## Copilot Code:

With 68 lines of code over 3 classes, the generated code has 22.6 lines per class. Running dart analyze finds 9 issues with 4 possible `const` declarations, 2 missing `key` parameters, 1 issue related to `createState()` and 2 uses of `print()`, which should be replaced with other forms of logging in production code.

automatische Übersetzung von  
Java => Go  
Angular => Flutter

# Ergebnisse Angular => Flutter



alle 3 Tools erfüllen die Aufgabe

- GPT liefert die Funktionalität der bestehenden Angular Anwendung und hält die Projektstruktur konsistent



# Ergebnisse Java (*Advent of Code* #3) => Go

mit allen 3 Tools gut machbar, mit Ausnahme

- Gemini liefert eine fehlerhafte Bedingung, da die Vorrangregeln der Operatoren in Go nicht beachtet werden

Statt

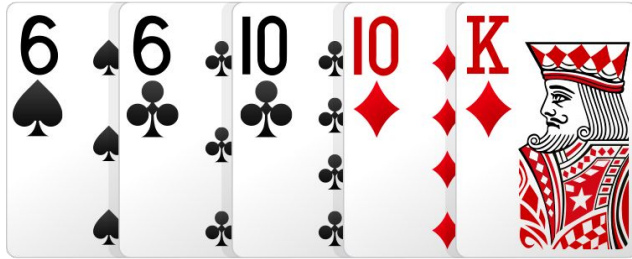
```
if char < '0' || char > '9' && char != '.' { ... }
```

muss die Bedingung korrekt wie folgt formuliert werden:

```
if (char < '0' || char > '9') && char != '.' { ... }
```

Beispiel: Code-Generierung 'from scratch'  
(Copilot in IntelliJ IDEA, Java)

# Texas Hold'em Poker, Beispiele für zwei 'Hände'



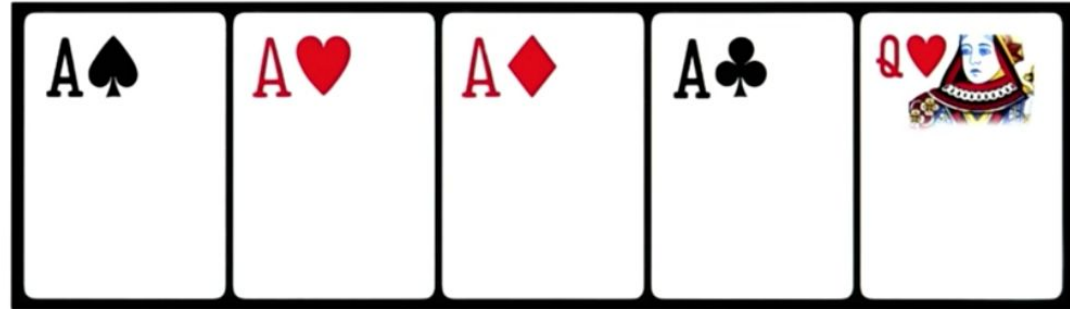
2 Paare



Full House

# Ass-Poker-Beschreibung im Texas Hold'em Jargon

FOUR ACES, AND A  
QUEEN KICKER!



# Zusammenfassung der Texas-Hold'em Regeln zur Bewertung einer Hand

Eine detaillierte Beschreibung der Texas-Hold'em Poker-Regeln findet sich zB auf der [Texas-Hold'em-Wikipedia](#)-Seite. Für die Bewertung einer Hand als Basis für den Vergleich zweier Hände ist insbesondere folgende Tabelle daraus relevant:

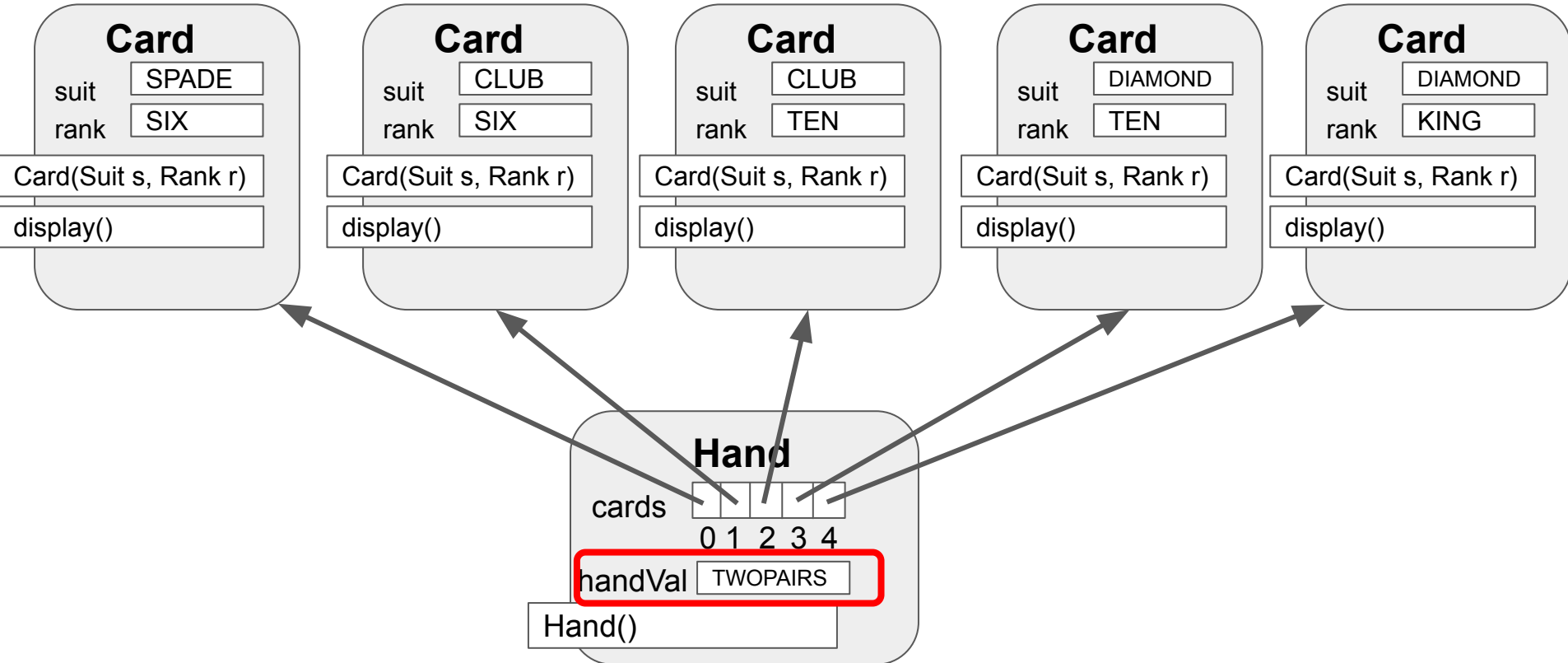
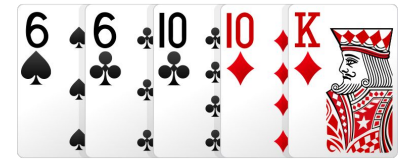
Hand	Kombinationen	Wahrscheinlichkeit	Beschreibung	Beispiel
Royal Flush	4 324	0,003232 %	Die Folge 10 bis Ass in einer Farbe	10♣ J♣ Q♣ K♣ A♣
Straight Flush	37 260	0,027851 %	Fünf aufeinanderfolgende Karten einer Farbe	5♠ 6♠ 7♠ 8♠ 9♠
Vierling, Poker, Four of a kind	224 848	0,168067 %	Vier Karten des gleichen Wertes	A♠ A♥ A♦ A♣
Full House	3 473 184	2,596102 %	Ein Drilling und ein Paar	K♥ K♦ K♣ 9♠ 9♦
Flush	4 047 644	3,025494 %	Fünf beliebige Karten einer Farbe	6♥ 8♥ Q♥ K♥ A♥
Straße, Straight	6 180 020	4,619382 %	Fünf aufeinanderfolgende Karten	7♥ 8♥ 9♠ 10♦ J♣
Drilling, Three of a kind	6 461 620	4,829870 %	Drei Karten des gleichen Wertes	Q♦ Q♥ Q♣
Zwei Paare, Two pair	31 433 400	23,495536 %	Zwei Paare	10♠ 10♥ A♥ A♣
Ein Paar, One pair	58 627 800	43,832255 %	Zwei Karten gleichen Wertes	J♣ J♠
High Card	23 294 460	17,411920 %	Die höchste einzelne Karte	A♥
Summe	133 784 560	100 %		

# Wie kann der Wert einer Hand geschickt ermittelt werden?

Die Aufgabe ist schwieriger, als es auf den ersten Blick ersichtlich ist:

- in manchen Fällen sind die Farben (Suit) ausschlaggebend (zB Flush), in manchen Fällen die Wertigkeit (Rank) der Karten (zB Paar, Drilling, Poker), bei einem Straight Flush beziehungsweise einem Royal Flush sind sowohl die Farbe als auch die Wertigkeit relevant
- in manchen Fällen müssen möglicherweise auch die "übrigen" Karten in absteigender Wertigkeit berücksichtigt werden: bei einer High-Card, einem Paar, zwei Paaren, einem Drilling, einem Poker
- bei einem Full House (Drilling und Paar) zählt bei einem Vergleich, wenn die andere Hand auch ein Full House ist, zuerst die Wertigkeit des Drillings, dann des Paares, auch wenn die Wertigkeit des Drillings niedriger als die des Paares ist
- bei einer Strasse kann ein Ass auch als Wert 1 genommen werden und so eine Strasse: Ass, 2, 3, 4, 5 gebildet werden

# ein Hand-Objekt mit 2 Paaren:



**Ausblick**



# Wie lange werden wir noch programmieren?

- Hypothese von Matt Welsh (PhD Computer Science UC Berkeley, Informatik Prof. Harvard, dann bei Google und Startups):
  - derzeit sind Programmiersprachen noch nötig, damit der Mensch einen Einblick hat
  - irgendwann wird dies nicht mehr nötig sein, und LLMs werden die Anwendungen unmittelbar beigebracht werden
  - in natürlicher Sprache, iterativ: 'Natural Language Computer'
- [LLMs and The End of Programming](#) (Harvard Univ., Oct. 2023)