Portfolioprüfung – Werkstück A – Alternative 6

Aufgabe 1 Aufgabe

Entwickeln und implementieren Sie ein Buzzword-Bingo-Spiel mit Interprozesskommunikation, mit dem Benutzer in der Shell gegeneinander Buzzword-Bingo spielen können.

Die Benutzer (mindestens zwei Spieler) sollen gegeneinander spielen können und der Spieler, der als erster das Buzzword-Bingo gelöst hat, hat gewonnen.

Buzzword-Bingo ist ein Klassiker der IT-Geschichte und hilft dabei, die übermäßige Verwendung inhaltsleerer Schlagwörter (häufig Anglizismen) in Vorträgen bzw. Präsentationen sichtbar zu machen. Der Beschreibung von Buzzword-Bingo auf Wikipedia ist nichts hinzuzufügen:

> Buzzword-Bingo, in der späteren Verbreitung auch Bullshit-Bingo und Besprechungs-Bingo genannt, ist eine humoristische Variante des Bingo-Spiels, die die oft inhaltslose Verwendung zahlreicher Schlagwörter in Vorträgen, Präsentationen oder Besprechungen persifliert.
>
> Statt Bingokarten mit Zahlen werden Karten mit Schlagwörtern (engl. buzzwords) benutzt. Im Gegensatz zum originalen Bingo, bei welchem die zu streichenden Zahlen aus einer Lostrommel gezogen werden, werden Wörter gestrichen, wenn sie genannt werden. Bei einer vollständig gefüllten Reihe, Spalte oder Diagonale soll der Spieler den Regeln nach aufstehen und *Bingo* oder auch, um die Inhaltsleere der Wortphrasen hervorzuheben, *Bullshit* rufen. Mit dem Spiel und diesem Ausruf wird gleichzeitig die übermäßige Verwendung oft inhaltsleerer Schlagwörter kritisiert.
>
> Quelle: `http://de.wikipedia.org/wiki/Buzzword-Bingo`

Zudem sehenswert: `https://www.youtube.com/watch?v=d5zRe8wa4pM`

Entwickeln und implementieren Sie Ihre Lösung als Bash-Skript oder als C-Programm als freie Software (Open Source) und verwenden Sie hierfür ein Code-Repository, z.B. bei GitHub.

Bearbeiten Sie die Aufgabe in Teams zu **5 Personen**.

Schreiben Sie eine aussagekräftige und ansehnliche Dokumentation (Umfang: **8-10 Seiten**) über Ihre Lösung.

Bereiten Sie einen Vortrag mit Präsentationsfolien und eine Live-Demonstration (Umfang: **15-20 Minuten**) vor. Demonstrieren Sie die Funktionalität der Lösung in der Übung.

Aufgabe 2 Anforderungen

- Das fertige Programm soll eine Kommandozeilenanwendung sein. Es soll möglich sein, das Spiel nur aus einer Shell heraus zu spielen. Es soll komplett in der Shell ablaufen!
- Der Quellcode soll durch Kommentare verständlich sein.
- Die Spieler (mindestens zwei Spieler) sollen gegeneinander spielen können.
- **Jeder Spieler ist ein eigener Prozess. Nutzen Sie zur Entwicklung und Implementierung beliebige Formen der Sychronisation von Prozessen, Interprozesskommunikation und Sychronisation von Prozessen wie Signalieren, Gemeinsamer Speicher, Nachrichtenwarteschlangen, Pipes, Sockets, Semaphoren, etc. Sie haben die freie Auswahl.**
- Jeder Spieler, der am Spiel teilnimmt, bekommt vom Spiel eine eigene Bingokarte in der Shell generiert.
- Die Anzahl der Felder in der Breite und Höhe der zu generierenden Bingokarte gibt der Benutzer, der das Spiel eröffnet, per Kommandozeilenargument an. Also z.B. `-xaxis 5 -yaxis 5`
- Die Werte der Felder auf der Bingokarte sollen aus einer Textdatei eingelesen und per Zufall verteilt werden. Konkret sollen die Benutzer, der das Spiel eröffnet, die Möglichkeit haben, per Kommandozeilenargument den Pfad und Dateinamen der Textdatei zu definieren, also z.B. `-wordfile <dateiname>`.
- Die Textdatei kann auch gerne mehr Wörter enthalten und es werden entsprechend so viele Wörter zufällig vom Buzzword-Bingo-Spiel aus der Datei importiert und auf der bzw. den Bingokarte(n) verteilt, wie es die definierte Höhe und Breite vorgibt.
- Idealerweise wählen die Spieler mit der Tastatur (und evtl. auch mit der Maus) einzelne Felder aus, um diese zu streichen (bzw. zu markieren). Alternativ ist

eine Auswahl der Felder durch Eingabe der Koordinaten mit der Tastatur auch denkbar.

- Hat ein Spieler oder ein Gegenspieler eine komplette Spalte, Zeile oder Diagonale der Bingokarte an Feldern gestrichen bzw. markiert, gilt das Spiel als gewonnen, was bei allen Spielern angezeigt wird. Das kann beispielsweise durch eine Laufschrift geschehen, durch ein Blinken oder durch ein Invertieren der Farben in der Shell, etc.

- Bei 5x5 oder 7x7 Feldern bleibt das Feld in der Mitte üblicherweise frei (*Joker*).

- Zur Dokumentation des Spiels soll das Programm für jedes Bingospiel (also für eine generierte Bingokarte) eine Logdatei mit folgendem Dateinamen anlegen:

  `YYYY-MM-DD-HH-MM-SS-bingo-SpielerNummer.txt`

  Die Felder stehen stellvertretend für Jahr (`YYYY`), Monat (`MM`), Tag (`DD`), Stunde (`HH`), Monat (`MM`) und Tag (`SS`) und müssen die jeweils aktuell gültigen Werte enthalten. Die Logdatei soll während des Spiels mit sinnvollen Daten (Zeilen) befüllt werden. Sinnvolle Beispiele sind:

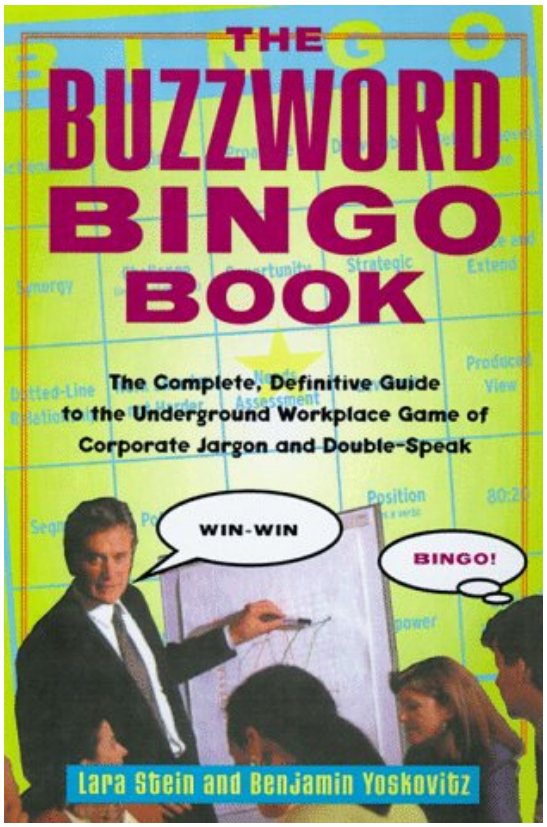
  - `YYYY-MM-DD-HH-MM-SS Start des Spiels`
  - `YYYY-MM-DD-HH-MM-SS Größe des Spielfelds: (X-Achse/Y-Achse)`
  - Eine Zeile für jedes gestrichene bzw. markierte Wort (in der Reihenfolge, in der es gestrichen bzw. markiert wurde) inklusive der Koordinaten des Worts auf dem Spielfeld in folgender Struktur:
    `YYYY-MM-DD-HH-MM-SS Wort (X-Achse/Y-Achse)`
  - `YYYY-MM-DD-HH-MM-SS Sieg` oder `Abbruch`
  - `YYYY-MM-DD-HH-MM-SS Ende des Spiels`

- Für die „grafische Darstellung“ und Bedienung in der Shell können Sie eine Bibliothek wie **ncurses**[1] [2] (für C-Programme), **Termbox**[3] (für C-Programme oder Python-Scripte), **dialog**[4] [5] [6] (für Shell-Scripte) oder **Whiptail**[7] [8] [9] (für Shell-Scripte). Zwingend nötig ist das aber nicht.

[1] `http://openbook.rheinwerk-verlag.de/linux_unix_programmierung/Kap13-002.htm`
[2] `https://de.wikibooks.org/wiki/Ncurses:_Grundlegendes`
[3] `https://github.com/nsf/termbox`
[4] `http://openbook.rheinwerk-verlag.de/shell_programmierung/shell_007_007.htm`
[5] `https://www.linux-community.de/ausgaben/linuxuser/2014/03/mehr-komfort/`
[6] `https://linuxkurs.spline.de/Ressources/Folien/Linux-Kurs-7.pdf`
[7] `https://en.wikibooks.org/wiki/Bash_Shell_Scripting/Whiptail`
[8] `https://saveriomiroddi.github.io/Shell-scripting-adventures-part-3/`
[9] `https://www.dev-insider.de/dialogboxen-mit-whiptail-erstellen-a-860990/`

Aufgabe 3 Einige Buzzwords zur Inspiration

- Synergie
- Rating
- Wert-schöpfend
- Benefits
- Ergebnisorientiert
- Nachhaltig
- Hut aufhaben
- Visionen
- Zielführend
- Global Player
- Rund sein
- Szenario
- Diversity
- Corporate Identitiy
- Fokussieren
- Target
- Benchmark
- Herausforderung(en)/Challenges
- Gadget
- Synergie
- Value
- Smart
- Web 2.0 oder 3.0
- Qualität
- Big Picture
- Revolution
- Pro-aktiv
- Blog
- Community
- Social Media
- SOA
- Skalierbar
- Return on Invest (ROI)
- Wissenstransfer
- Best Practice
- Positionierung/Positionieren
- Committen
- Geforwarded
- Dissemination
- Skills
- Gap
- Follower
- Win-Win



Bildquelle: `http://www.brokenwire.net`

Aufgabe 4 Literatur

- Foliensätze 4 und 6 der Vorlesung **Betriebssysteme und Rechnernetze** im SS2022
- **Betriebssysteme kompakt**, *Christian Baun*, 2. Auflage, Springer Vieweg, S. 200-252
- **Betriebssysteme**, *Erich Ehses, Lutz Köhler, Petra Riemer, Horst Stenzel, Frank Victor*, 1. Auflage, Pearson (2005), S. 55-84
- **Betriebssysteme**, *Carsten Vogt*, 1. Auflage, Spektrum (2001), S. 109-127
- **Betriebssysteme**, *William Stallings*, 4. Auflage, Pearson (2003), S. 334-339