SPU·FEM
Fakulta
ekonomiky
a manažmentu

Programovanie

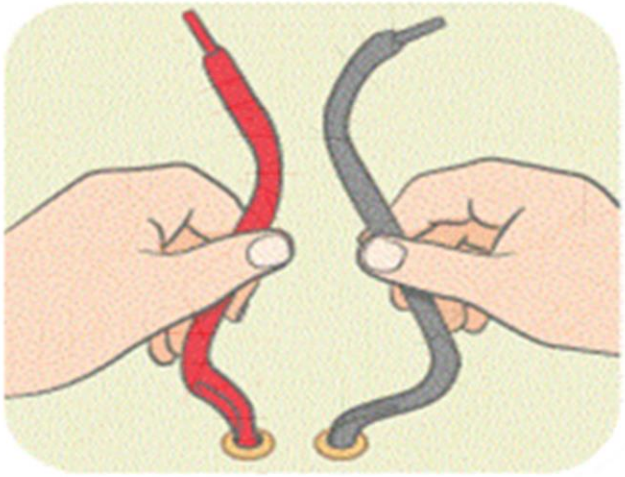
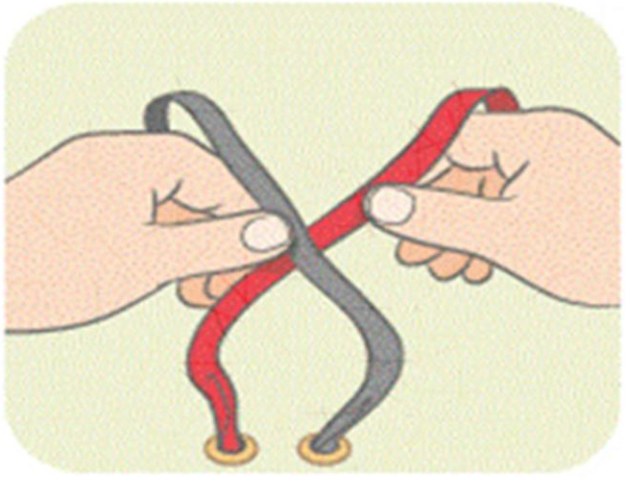
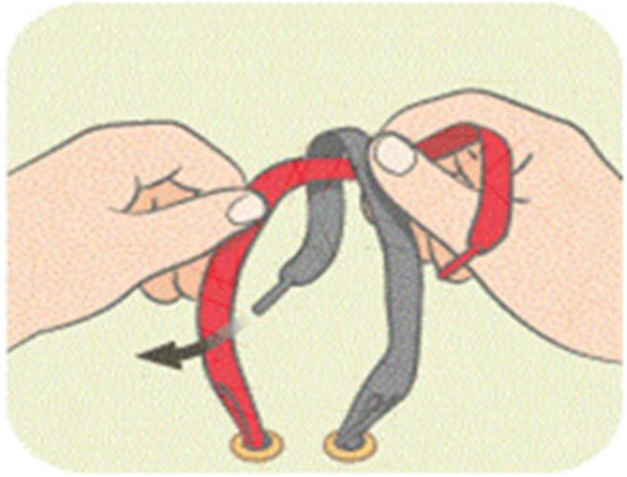
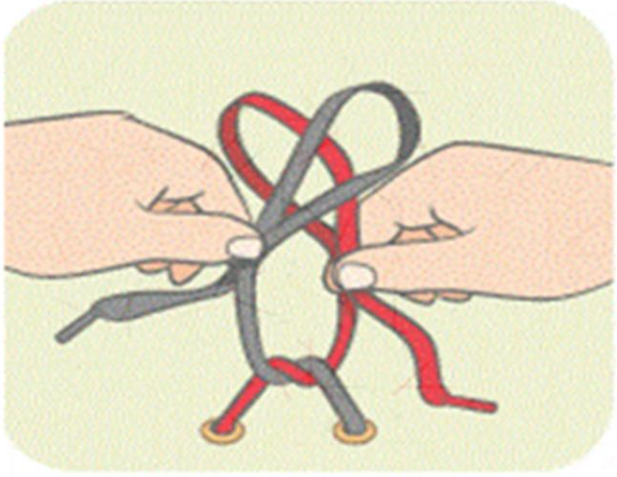
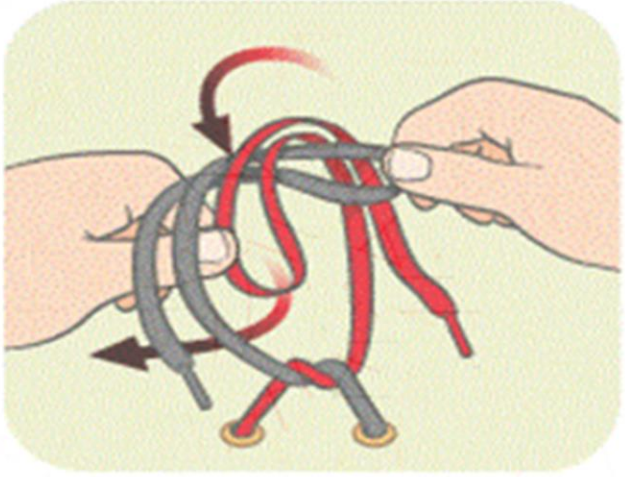
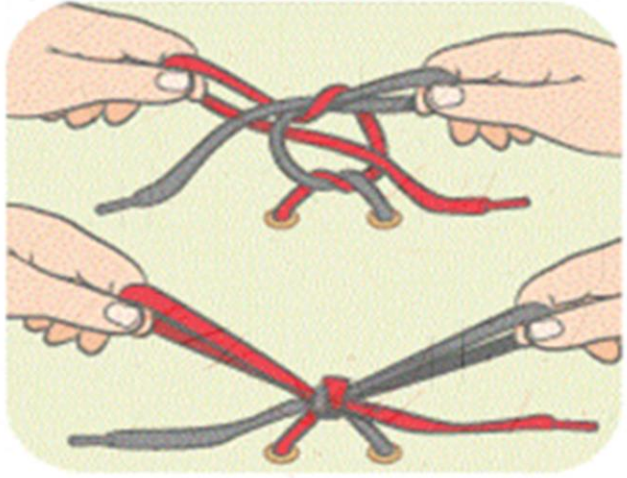
RNDr. Tomáš Tóth, PhD.

Čo sa dnes dozvieme

- Čo je to algoritmus a algoritmizácia
- Aké sú vlastnosti algoritmu
- Čo je to programovanie a programovací jazyk
- Na čo je dobré sa učiť programovanie
- Riadiace štruktúry v programovaní

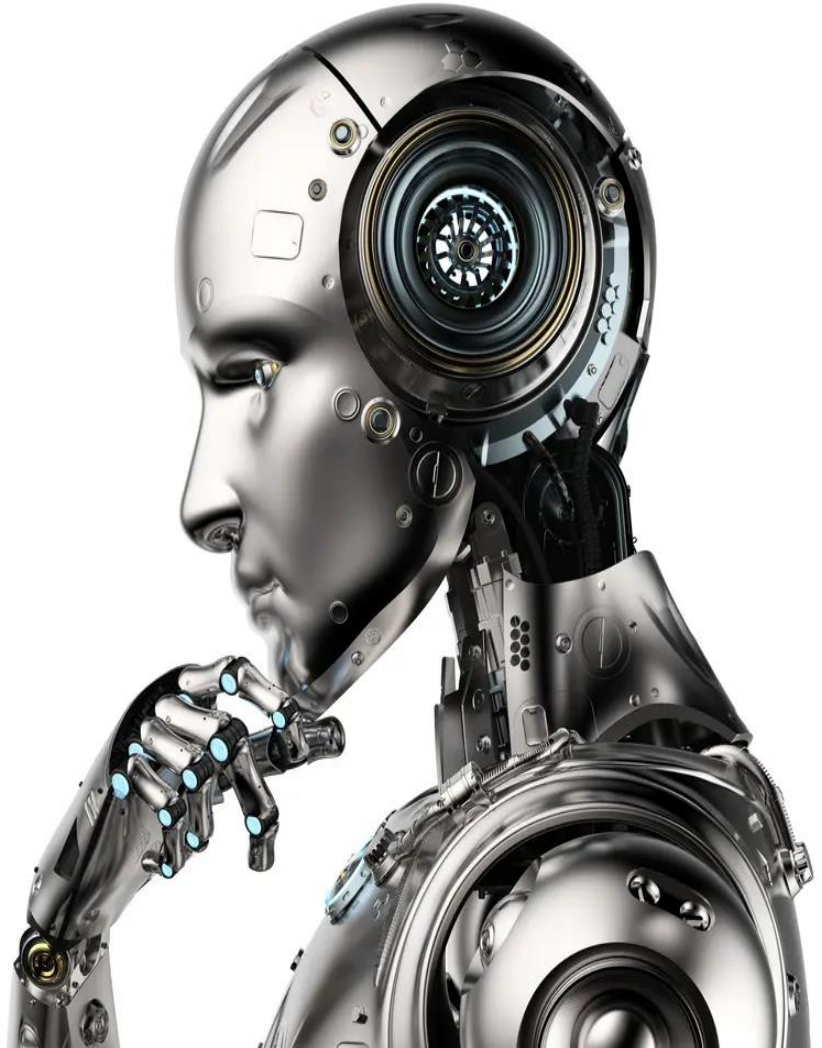
Na počiatku bol ...

Príklad: Problém viazania šnúrok

Človek vs. stroj

- Počítaču nestačí zostručnený postup
- Počítač si zakaždým pozrie postup a bude sa podľa neho riadiť pri riešení problému
- Človek vykonáva niektoré činnosti už automaticky, nezamýšľa sa už nad postupom
- Počítač uvažuje v absolútnej presnosti nad postupom v každom jednom momente nad každým jedným krokom každého postupu do úplného detailu



Príklad: Problém pitia kakaa

1. Overíme dostatok surovín
2. Do hrnčeka nasypeme kakao
3. Dáme zohriať mlieko
4. Skontrolujeme teplotu mlieka - ak nie je dosť teplé vrátime sa do bodu 3
5. Kakao v hrnčeku zalejeme zohriatym mliekom
6. Necháme vychladnúť
7. Vypijeme

A je po probléme...

Myslieť ako počítač

- Ľudia používajú intuíciu, vedia si domyslieť a konať už aj na základe predtým naučeného
- Počítaču je potrebné presne určiť, čo má v každom momente a v každej situácii urobiť – potrebné do detailov rozpísať
- Cieľ:
  - Naučiť sa **„rozbiť“ každú jednu činnosť** na drobné najmenšie časti a vedieť si predstaviť v každom jednom momente **všetky možnosti, ktoré môžu nastať**
  - Naučiť sa premyslieť si **reakcie na každú z možností** a znovu ich popísať do najmenších detailov

Algoritmus a problém

- **Problém**
  - stav, v ktorom existuje rozdiel medzi tým, čo v danom momente máme a tým, čo chceme dosiahnuť
- **Riešenie problému**
  - odstraňovanie rozdielu medzi aktualnym stavom a tým, čo chceme dosiahnuť
- **Algoritmus**
  - postup, ktorým sa pri riešení problému riadime

Definícia algoritmu

Algoritmus je návod
(presná postupnosť krokov a inštrukcií)
na vykonanie činnosti, ktorý nás
od (meniteľných) vstupných údajov
privedie v konečnom čase
k výsledku

Vykonávanie činnosti na základe algoritmu označujeme ako ***výpočet***

Vlastnosti algoritmu

- Elementárnosť
- Determinovanosť
- Rezultatívnosť
- Konečnosť
- Hromadnosť
- Efektívnosť

Elementárnosť

- Postup je zložený z **jednoduchých** krokov, ktoré sú pre vykonávateľa (počítač, nemysliace zariadenie, človek) **zrozumiteľné**
- Napr. príkazy:

  „Krájaj týždeň starú kapustu!“

  „Pridaj cukor“

Determinovanosť

- Postup je zostavený tak, že v každom momente jeho vykonávania je **jednoznačne určené, aká činnosť má nasledovať**, alebo či sa už postup skončil
- Príklad: umyť – vyzliecť – spať
  - Ak by sa nevykonalo v tomto poradí, nemuselo by to byť tým, čo bolo pôvodne zamýšľané
  - Ak by si ich počítač vysvetlil po svojom, mohlo by sa stať, že najprv pôjde spať (vykoná časovo najnáročnejšiu úlohu) a až keď sa vyspí, vyzlečie sa a umyje

Rezultatívnosť

- Realizácia dokázateľne **vedie po konečnom počte krokov k správnemu výsledku** pri riešení ľubovoľnej úlohy zo skupiny úloh, pre ktorú bol algoritmus vytvorený
- Skupina problémov → návrh algoritmu riešenia → riešenie hociktorého problému zo skupiny v ľubovoľnom čase → vždy rovnaký výsledok
- V bežnom živote sa to vždy podariť nemusí – varenie/pečenie
  - Raz viac slané, inokedy menej slané
  - Raz viac upečené, inokedy menej upečené

Konečnosť

- Algoritmus **musí mať vždy svoj koniec** (musí skončiť)
  - človek, pracujúci s problémom na základe skúseností dokáže určiť, či jeho postup dá alebo nedá správny výsledok (resp. či skončí alebo nie)
  - počítač bez skúseností sa na takejto úrovni rozhodnúť nedokáže

- Algoritmus pre počítač:

  1. polož hrniec s jedlom na varič
  2. pusti plyn
  3. miešaj, kým nezačne vrieť

  medzi tým sa vypne plyn ...

Hromadnosť

- Postup je **použiteľný na celú triedu** prípustných vstupných údajov
  - Nerieši len jednu konkrétnu situáciu
  - Postup je všeobecne popísaný tak, aby riešil ktorúkoľvek úlohu daného typu
- Napr. výpočet obsahu štvorca – nie iba jedného konkrétneho, ale ľubovoľného s rôznymi rozmermi (dĺžkami strán)
- Nepovinná vlastnosť, ale užitočná

Efektívnosť

- Opäť nepovinná vlastnosť
- Navrhnúť taký **postup, ktorý s použitím minimálnych prostriedkov v čo najkratšom čase vyrieši problém**
- Aj neefektívny algoritmus je algoritmom
- Dôležitá najmä pri spracúvaní veľkého množstva údajov

Proces, ktorý vykonávame pri zápise algoritmu sa nazýva ***algoritmizácia***

Na čo mi je algoritmizácia?

- Život je algoritmus – rôzne činnosti majú svoj postup
- Vďaka popisu **dokážeme** vykonávaním algoritmu **poveriť iného človeka alebo počítač**
- Vďaka vyjadreniu myšlienok sa nám **problém stáva zrozumiteľnejším** a sme schopní lepšie mu porozumieť, vylepšiť ho

Jazyk ako dorozumievací prostriedok

- Jazyk pôvodne ako prostriedok komunikácie iba medzi ľuďmi, ale dnes už aj medzi človekom a strojom

Použitie ľudského jazyka je problematické

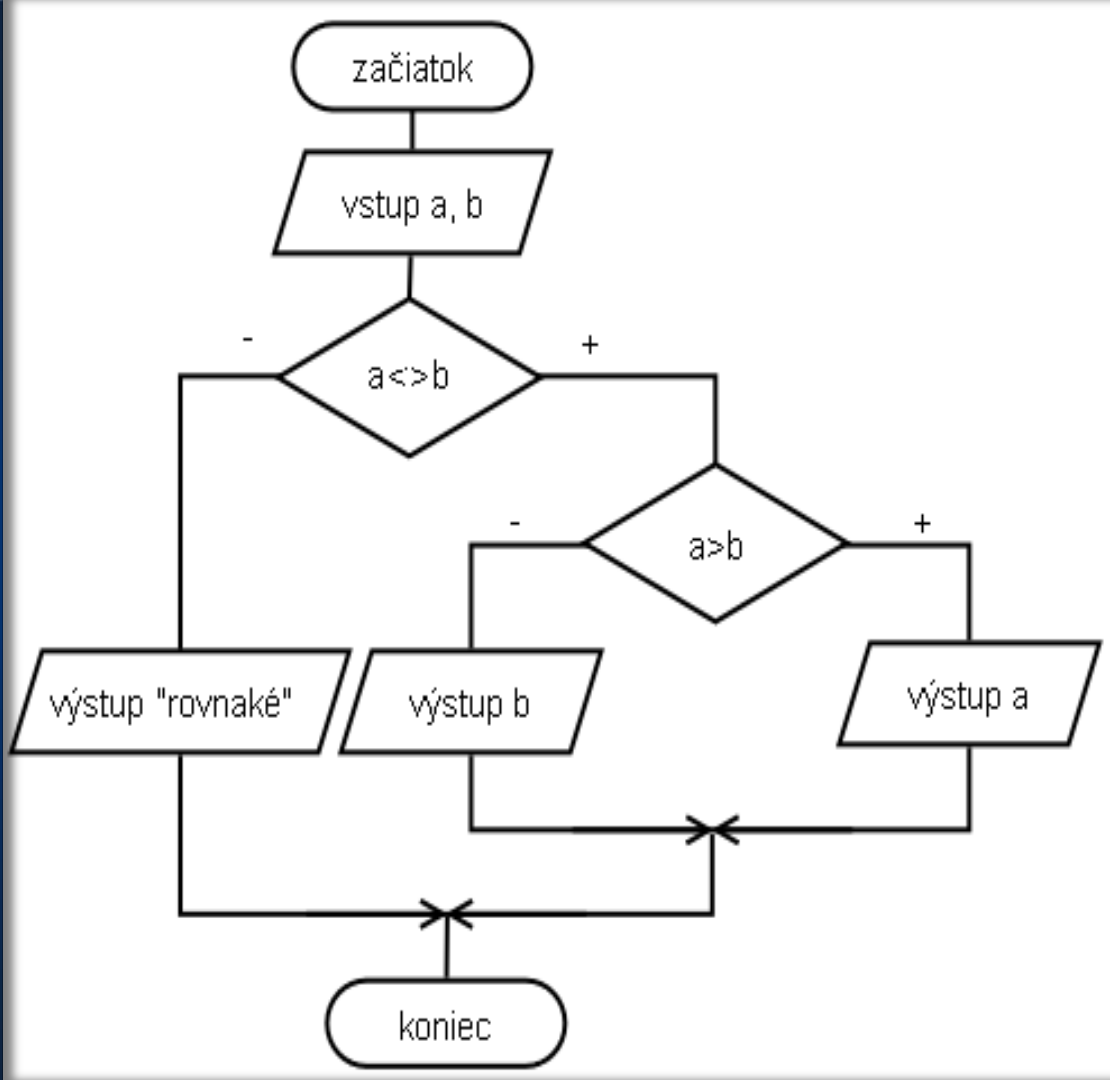
- Veľké množstvo slov
- Slová vznikajú a zanikajú
- Množstvo výnimiek
- Množstvo tvarov (pády, časy, neurčitok)
- Synonymá (rovnoznačné slová – urobiť/vykonať)
- Homonymá (rovnozvučné slová – koruna (stromu, peniaz))

Algoritmický jazyk

- Algoritmický jazyk:
  - **stabilný a nemenný zoznam slov umožňujúci presnú špecifikáciu príkazov**
  - je vyžadovaná **presnosť, konkrétnosť a adresnosť**
- Potreba redukcie prirodzeného jazyka na úzku skupinu slov, pomocou ktorých je možné požadovanú činnosť popísať a špecifikovať jej parametre

Algoritmický jazyk

- Napr. vývojé diagramy a ďalšie



Programovací jazyk

- Patrí medzi algoritmické jazyky
- **Formalizuje algoritmický jazyk** do zápisu, **ktorý dokáže spracovať a zrealizovať počítač** (alebo iné zariadenie)
- Často založené na redukcii slov anglického jazyka

Programovanie

Proces algoritmizácie s využitím programovacieho jazyka

Programovanie

Proces vytvárania sady inštrukcií, ktoré počítač vykonáva na dosiahnutie určitého cieľa

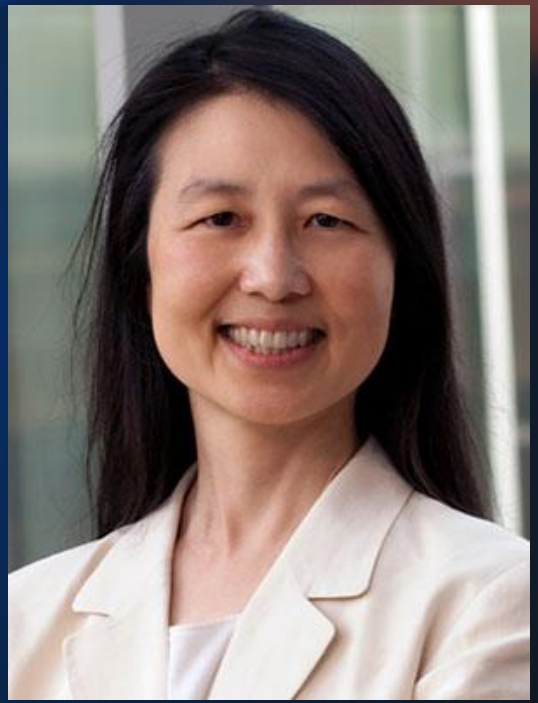
Prečo sa učiť programovanie

Aby sa človek stal programátorom a dokázal tvoriť softvér (desktopové aplikácie, mobilné aplikácie, webové aplikácie)

Prečo sa učiť programovanie – nie len pre programátorov

- Zlepšenie digitálnych zručností
- Lepšie porozumenie svetu technológií a fungovaniu technických zariadení
- Rozvoj zručností dôležitých v 21. storočí:
  - Zručnosť riešenia problémov
  - Informatické myslenie
  - Kritické myslenie
  - Algoritmické myslenie
  - Kreatívne myslenie
  - Analytické myslenie
  - Logické myslenie
  - Abstraktné myslenie
  - Deduktívne myslenie
  - Schopnosť spolupracovať

Schopnosť “myslieť ako informatik” by mala patriť medzi základné zručnosti, akými sú čítanie, písanie a počítanie.



Jeannette M. Wing

- Profesorka informatiky na Columbia University
- Zastávala a aj stále zastáva rôzne vedúce pozície v akademickej oblasti a aj v komerčnej sfére

Životný cyklus tvorby programu

- **Definícia problému**
  - Jasná definícia problému – čo je potrebné vyriešiť
- **Analýza problému**
  - Delenie problému na menšie a jednoduchšie podproblémy, definovanie vstupov a výstupov programu
- **Návrh riešenia**
  - Popis postupnosti krokov ľudským/algoritmickým jazykom
- **Tvorba kódu**
  - Programovanie – prepis návrhu do zdrojového kódu
- **Testovanie**
  - Odstraňovanie chýb programu

Ada Lovelace

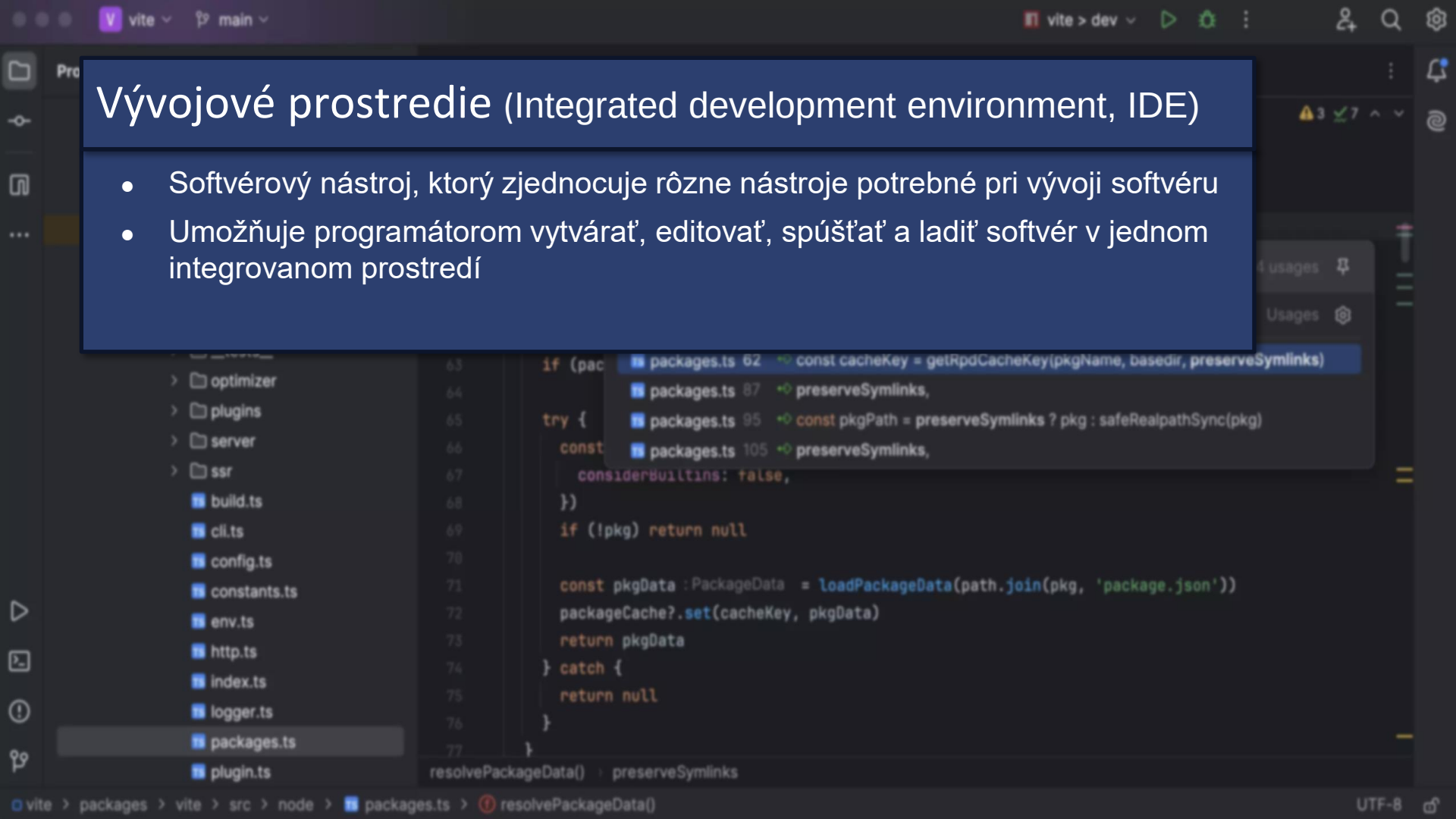
- Nar. 10.12.1815
- Grófka z Lovelace
- Britská matematička
- Napísala prvý program pre analytický stroj Charlesa Babbagea

Z histórie programovania - dierne štítky

Programovanie vytváraním rôznych kombinácií dier na štítku

Vývojové prostredie (Integrated development environment, IDE)

- Softvérový nástroj, ktorý zjednocuje rôzne nástroje potrebné pri vývoji softvéru
- Umožňuje programátorom vytvárať, editovať, spúšťať a ladiť softvér v jednom integrovanom prostredí

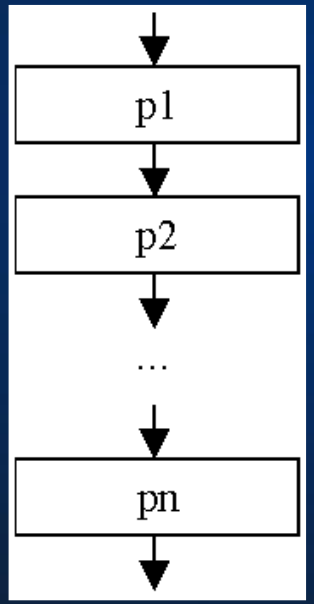


Riadiace štruktúry

- **Riadia sled vykonávania programu**
- Patrí sem:
  - Sekvencia
  - Vetvenie
  - Cyklus

Sekvencia

- Postupnosť príkazov vykonávaná **v** takom **poradí, v akom** sú jednotlivé časti **zapísané**

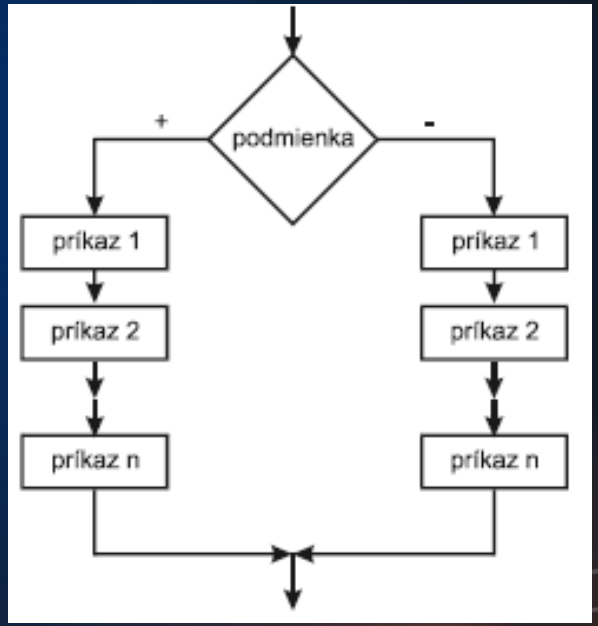


Zápis sekvencie vo vývojovom diagrame

p1, ..., *pn* sú príkazy
vykonajú sa v poradí, v akom sú zapísané

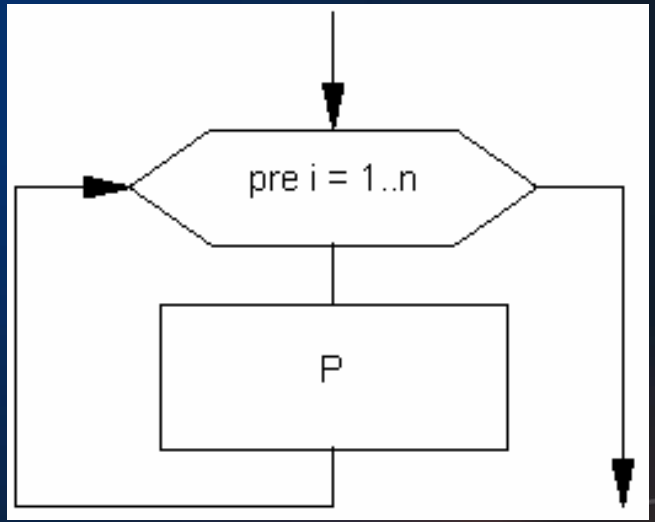
Vetvenie

- Poskytuje možnosť **rozhodnúť sa podľa pravdivosti skúmaného znaku**
- Skladá sa z podmienky uvedenej za slovíčkom *ak* a z príkazov, ktoré sa vykonajú v prípade kladného a záporného výsledku. Týmto dvom častiam hovoríme **vetvy**
- Ak je podmienka splnená –> vetva „+“
- Ak nie je podmienka splnená –> vetva „-“



Cyklus

- Zabezpečuje **opakované vykonávanie príkazov**,
  až kým nie je splnená podmienka

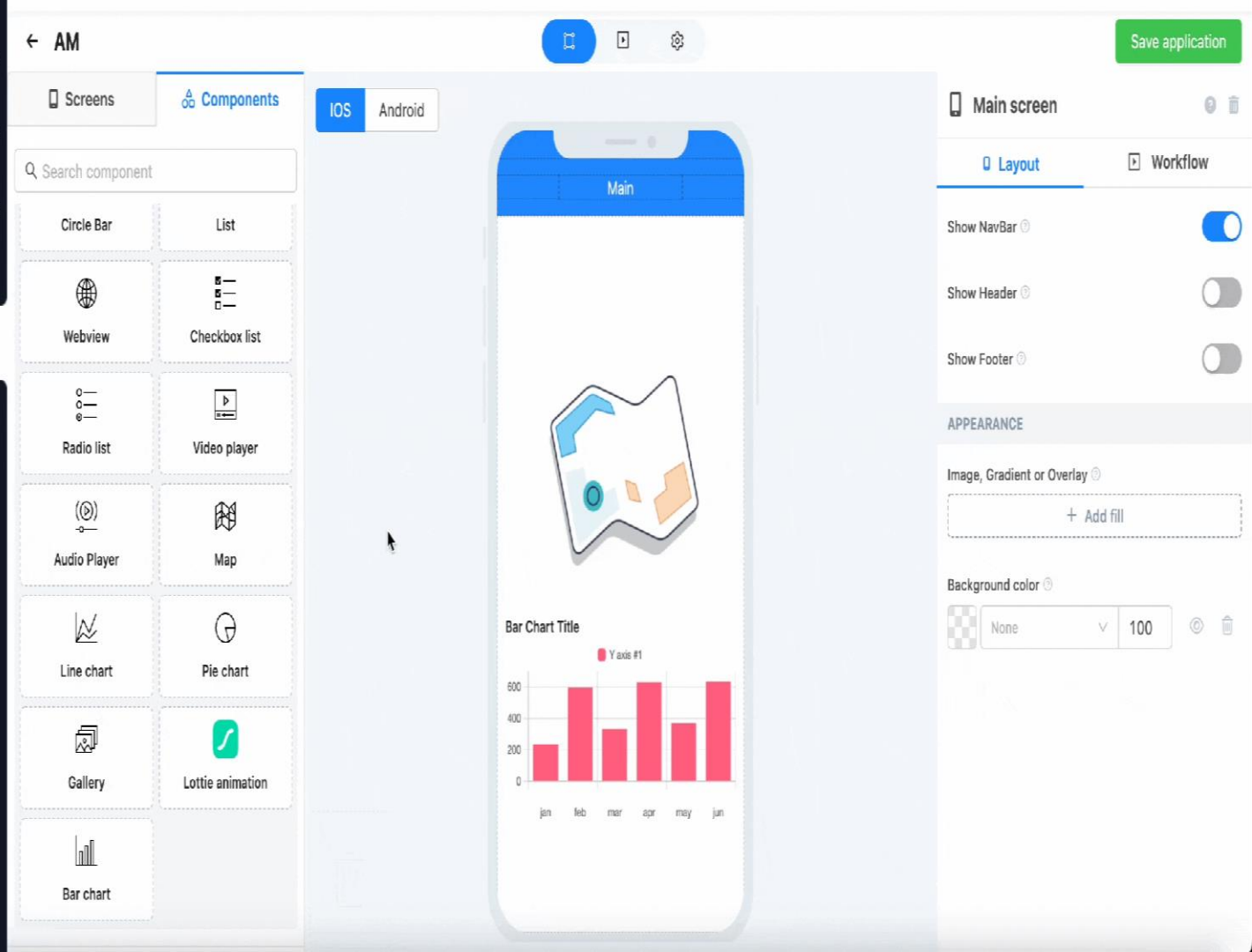


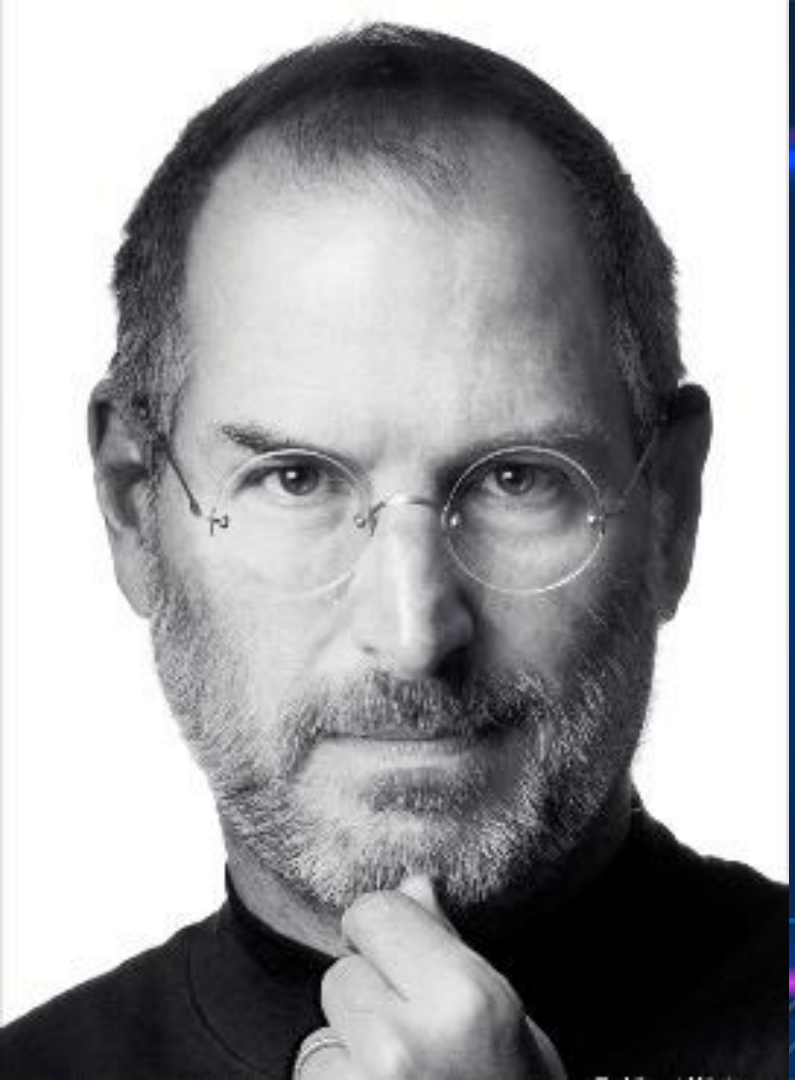
Softvérové inžinierstvo

- Disciplína, ktorá **sa zaoberá systematickým prístupom k návrhu, vývoju, prevádzke a údržbe softvéru**
- Využíva poznatky z informatiky, projektového manažmentu a ďalších oblastí
- Cieľom je vhodnými metódami vytvárať softvér, ktorý je spoľahlivý, efektívny a spĺňa požiadavky používateľov

Low Code, No Code ako budúcnosť programovania?

- Dva prístupy k tvorbe softvéru
- Zjednodušenie procesu vytvárania softvéru tým, že minimalizujú alebo úplne odstránia potrebu písania kódu
- Aplikácie sa vytvárajú prostredníctvom vizuálneho programovania a metódou „drag & drop“





„Everybody in this country should learn to program a computer...
because it teaches you how to think.“

Steve Jobs, co-founder of Apple Inc.


ĎAKUJEM
ZA POZORNOSŤ