9. Datové struktury, primitivní a strukturované datové typy

# Primitivní datové typy

* int (32bit, int32)
* long (64bit, int64)
* float (32bit, float32, může uchovat i větší čísla než int, ale s adekvátní ztrátou přesnosti)
* double (64bit, float64, chová se stejně jako float, ale má 2x přesnost)
* boolean (8bit, uchovává ale pouze hodnotu 1b)
* byte (8bit)
* short (16bit)
* char (16bit, v některých jiných jazycích se používá 8bit)

# Strukturované datové typy

* Array (pevně alokované místo v paměti)
  + Velikost v B = (počet prvků) \* (velikost prvku v B)
  + Může obsahovat pouze prvky stejného datového typu
* String
  + ,,Pole pro písmena“. Ve skutečnosti to ale tak jednoduché jako pole pro písmena není, protože máme různá kódování a u některých z nich není přímá závislost mezi písmenem a délkou jako například v UTF8, tak se jedná spíš o objekt, obsahující pole bytu (ne charu), který musí být přečten určitou další logikou
  + Konverzí stringu na char array může potenciálně dojít k chybě, pokud se budeme snažit převést znak s 3 a více bytovou reprezentací

# Datové struktury

* Linked list
  + Nemá pevně definovanou velikost, protože je založen na propojování objektů
  + Objekty linked listu obvykle nejdou v paměti za sebou, ale jsou různě rozházeny
  + Může uchovávat objekty různých tříd i různé datové typy (závisí na implementaci)
* Binary tree
  + V podstatě linked list, ale větví se
  + Do větvení lze zakomponovat logiku, tak že může sloužit k vyhledávání
* Graph
  + Opět se jedná v podstatě o linked list, který ale uchovává závislosti a může být tedy různě provázaný

# Datové struktury s logikou

* Stack
  + Může být implementován jak pomocí linked listu, tak pomocí pole
  + Pointa zásobníku je, že to, co šlo jako poslední dovnitř musí jít jako první ven (LIFO)
* Queue
  + Podobná implementace jako stack
  + Hlavní myšleka frony je, že to, co jde první dovnitř jde i jako první ven (FIFO)