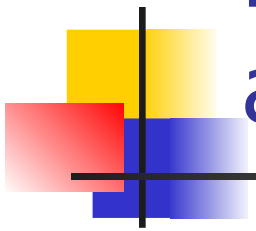
More practice, CSV and memory allocation



Schedule

- Sets intersection
- memory allocation
- CSV file handling
- structures (struct)

souce files:

`actors*.csv`

`CSV-read-write-append-actors.cpp`

Sets intersection (desctiption)

Read set **A** containg **N** elements, and set **B** containing **M** elements of integers, then evaluate their intersection. Handle possible errors:
- multiple ocassion of one element
- non integer value of a element

Sets intersection (notes)

memory allocation for a set **A** with **N** elements

```
int *arrayA = new int[N];
```

check validity of a new element

```
bool isInSet(int item, int arry[], int size){
```

eveluate the intersection concurrently

as you read the set **B**

take a care for correct output:

{} versus {, }

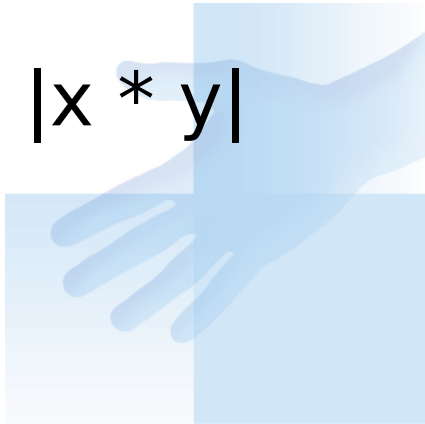
(use logical variable: , hodnota)

Cosine Measure of Similarity

N elements of vector, **double** data type

```
double *vctr1 = new double[N];
  // read & check possible wrong input
  // do the same for vctr2

the proper allocaded memory return is
checked by Progtest
  // delete [] vctr1;
  // any time, even in the wrong entry
  // then evaluate cosine similarity
  // perfect: write functions for |x|, |y|, |x * y|
  // real programmer:
  // ...but one loop could evaluate
  //    all three expressions as well :-)
```



CSV file open (read from)

open

```
string name = "actors.csv";
ifstream iF(name.c_str());
if (iF == NULL) {
   cout << "CSV file open error" << endl;
   return -1;
}
```

loop (read all entry/rows)

```
Actor actor;
 while (!iF.eof()) {
     readActor(iF, actor);
     writeActor(cout, actor);
 }
 iF.close();
```

CSV file read (from) structure

čtení

```
bool readActor(ifstream &fromStream,
               Actor &a) {
    getline(fromStream, a.name, ',');
    getline(fromStream, a.surname, ',');
    getline(fromStream, a.born, ',');
    getline(fromStream, a.address, ',');
    getline(fromStream, a.eye, ',');
    getline(fromStream, a.hair, '\n');
```

případně oddělovač: `';'`

CSV file write (into)

write a structure into CSV

```
bool writeActorToCSV(ofstream &toStream,
    Actor a) {
  toStream << a.name << ", "
           << a.surname  << ", "
           << a.born  << ", "
           << a.address  << ", "
           << a.eyes << ", "
           << a.hair << endl;
```

CSV file open (write into+app mode)

open (+ append)

```
ofstream oF;
oF.open(name.c_str(), ios::app);
if (oF == NULL) {
   cout << "CSV file open error" << endl;
   return -1;
}
```

CSV file write (into)

(function) write

```
bool writeActorToCSV(ofstream &toStream,
     Actor a) {
   toStream << a.name << ", "
   << a.surname  << ", "
   <<  a.born  << ", " << a.address
   << ", " <<  a.eye  << ", "
   << a.hair << endl;
 return true;
}
// ...function call:
   Actor actor;
   writeActorToCSV(oF, newA);
   oF.close();
```