HIGHER SCHOOL OF ECONOMICS

NATIONAL RESEARCH UNIVERSITY

Introduction to Programming

Structures and All, All, All

Sergey Shershakov

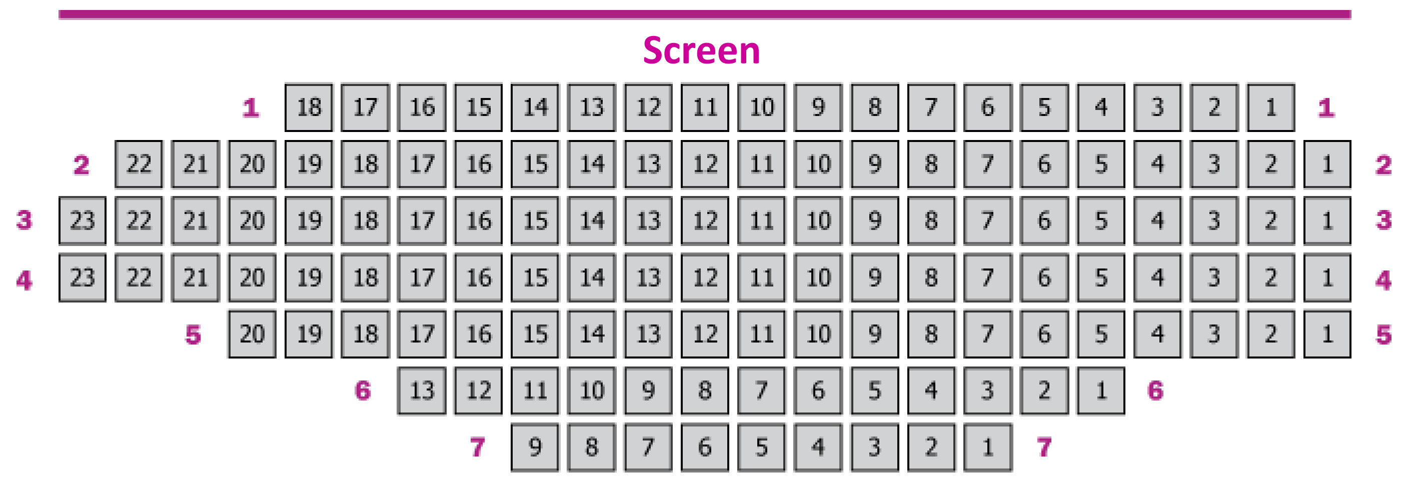
#7/30 Jan 2019

'cos there is no test today!

Contest 2 Problem 16

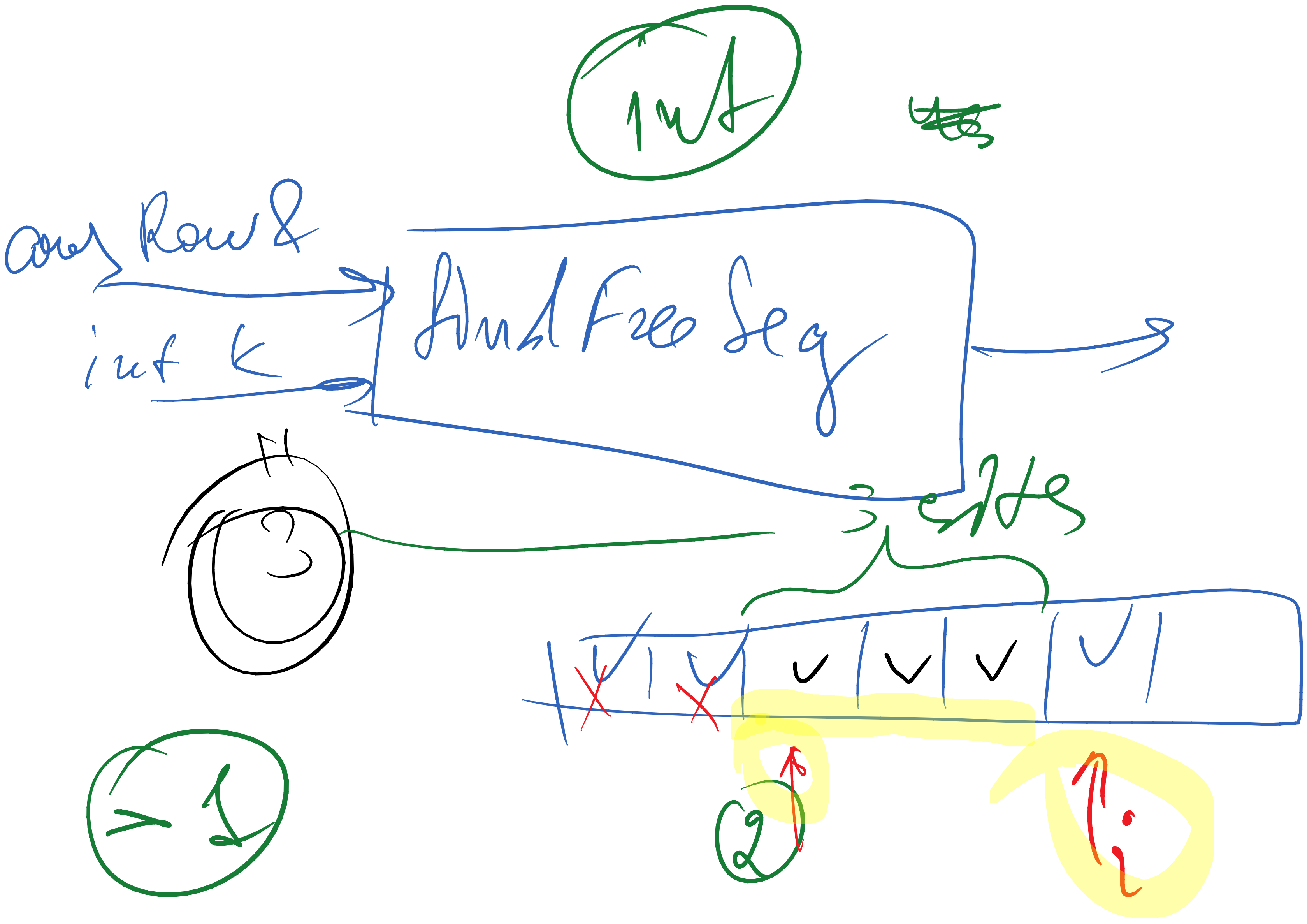
Let's Go to a Cinema!

x ,



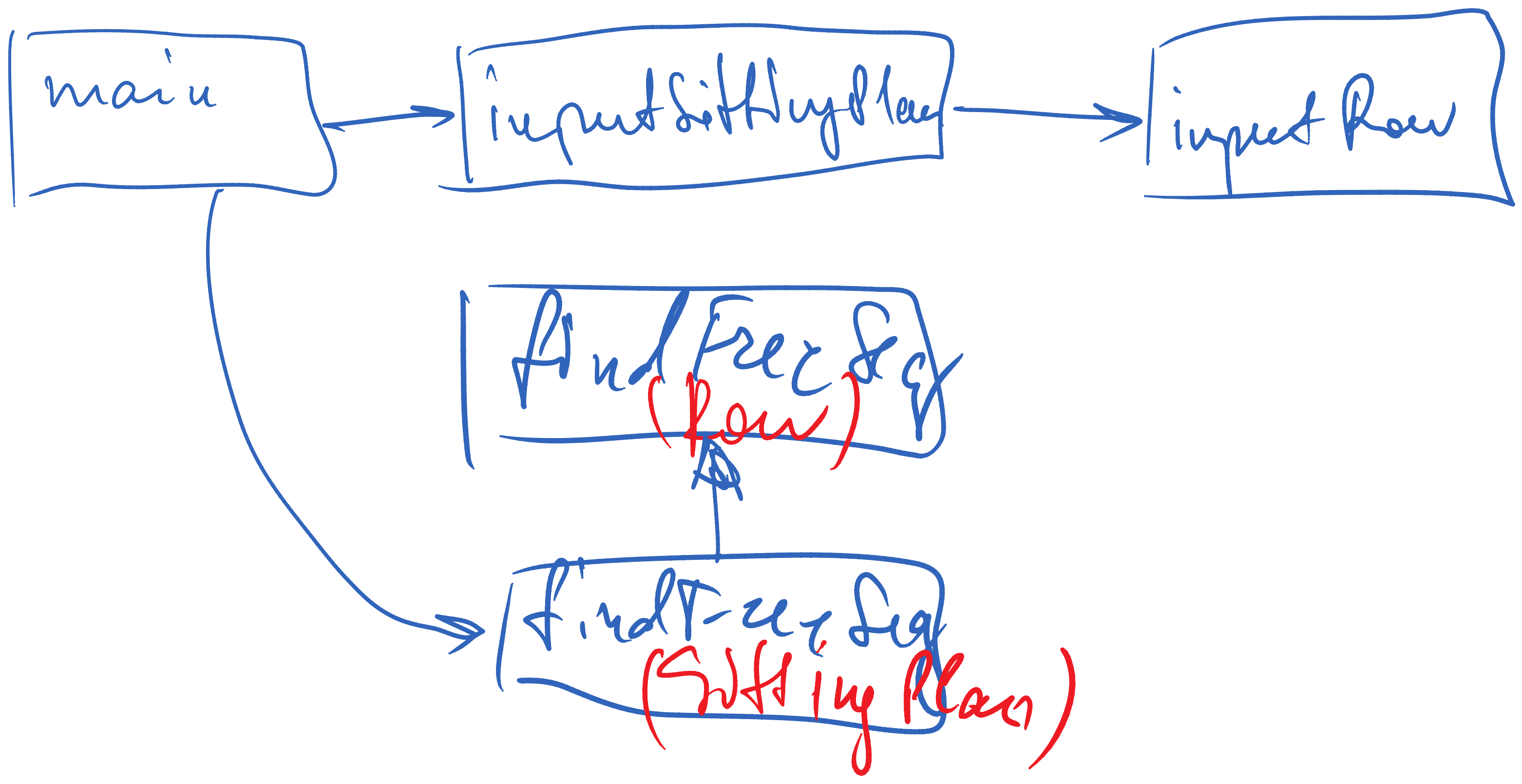
Jagged Array

1) input data: m rows, n_i seats for each i-th row; 1 — the seat is sold, 0 — the seat is free;
2) print data in a different format: a row per line, * is for sold seats, . is for free; sold/total ratio in the end of each row/line;
3) someone would like to buy k adjacent seats in the same row; one needs to determine whether it is possible or not;
4) how to modify the printing method for highlighting the free k seats by using "XXXX" notation?

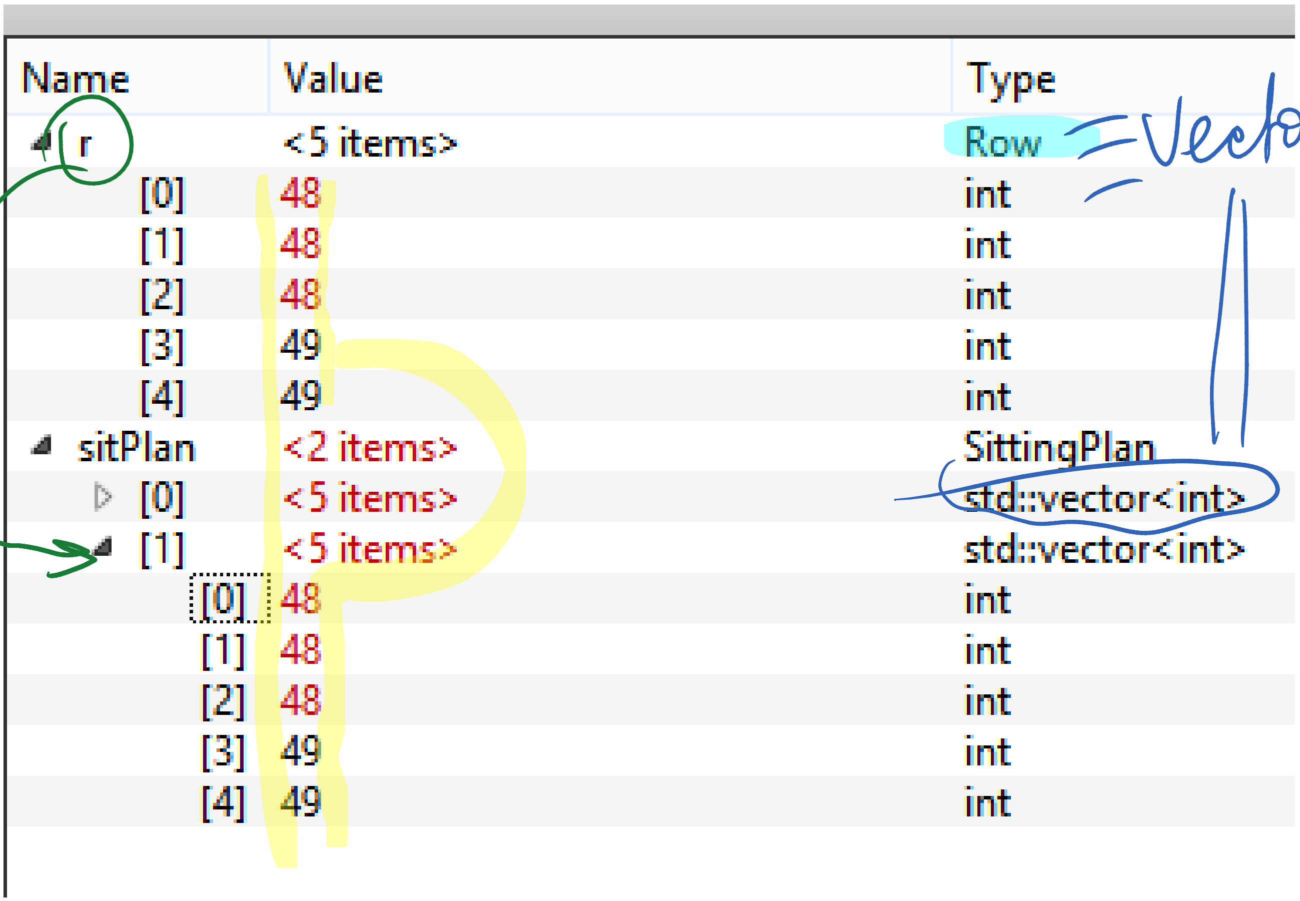


int
~~yes~~
arr Row&
int k
FindFreeSeq
3
3 elts
>1
2

Function Call Diagram



Object Copying



Name	Value	Type
r	<5 items>	Row
[0]	48	int
[1]	48	int
[2]	48	int
[3]	49	int
[4]	49	int
sitPlan	<2 items>	SittingPlan
[0]	<5 items>	std::vector<int>
[1]	<5 items>	std::vector<int>
[0]	48	int
[1]	48	int
[2]	48	int
[3]	49	int
[4]	49	int

a copy

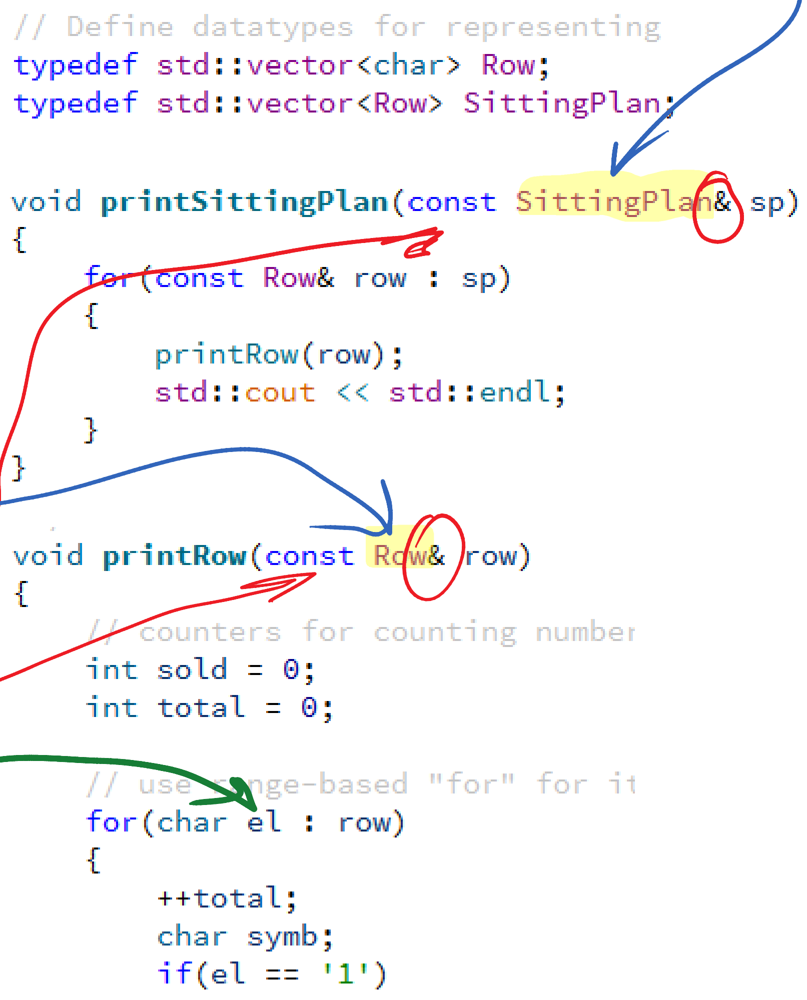
= vector<int>

Thumb-Rules on Using *Refs* and *Const Refs*

1. If a method has a parameter represented by a *complex object* (e.g., any one bigger than any POD such `int`, `double`, `T*`), and the method does not need a copy of an object represented by the parameter, it is better to pass the parameter by using *reference* or *const reference*.
   - POD types are more efficient to be passed *by value*.

2. Let a method have a parameter given by a *reference*, and let the method not change the value of the parameter, then the parameter must be given using a **const** ref.

3. If an object (e.g., a parameter) of a method is a *const* (*ref*), all derivatives of the object (its member fields, member methods and so on.) are also *const*.



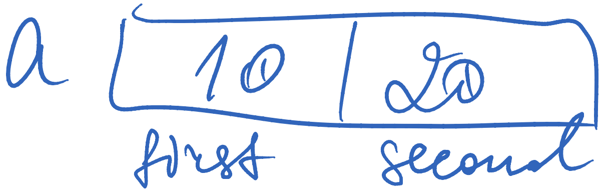
```
// Define datatypes for representing
typedef std::vector<char> Row;
typedef std::vector<Row> SittingPlan;

void printSittingPlan(const SittingPlan& sp)
{
    for(const Row& row : sp)
    {
        printRow(row);
        std::cout << std::endl;
    }
}

void printRow(const Row& row)
{
    // counters for counting number
    int sold = 0;
    int total = 0;

    // use range-based "for" for it
    for(char el : row)
    {
        ++total;
        char symb;
        if(el == '1')
```

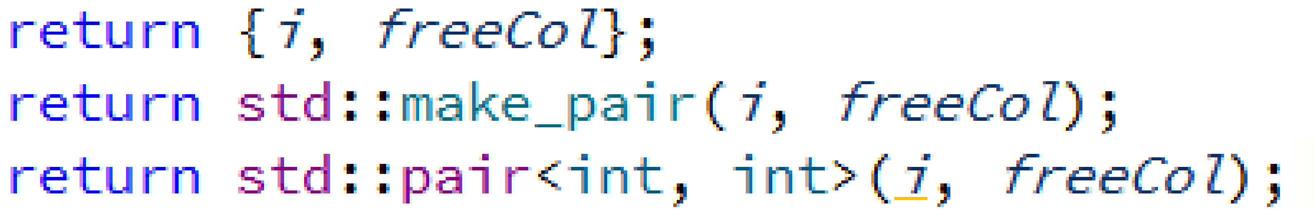
The `std::pair` Utility Class

- Simple structure representing a pair of objects that can have a different type

```
std::pair<Type1, Type2>
```

```
pair<int, int> a(10, 20);
a.first == 10;
a.second == 20;
```





```
return {i, freeCol};
return std::make_pair(i, freeCol);
return std::pair<int, int>(i, freeCol);
```

http://www.cplusplus.com/reference/utility/pair/pair/
https://en.cppreference.com/w/cpp/utility/pair

STRUCTURES

Structure as a Compound Type

the `struct` keyword

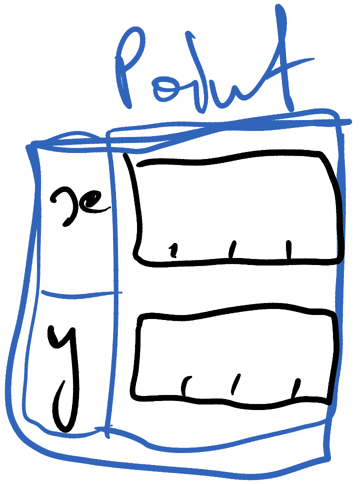
the name for a new (custom) type

```
struct Point
{
        int x;
        int y;
};
```

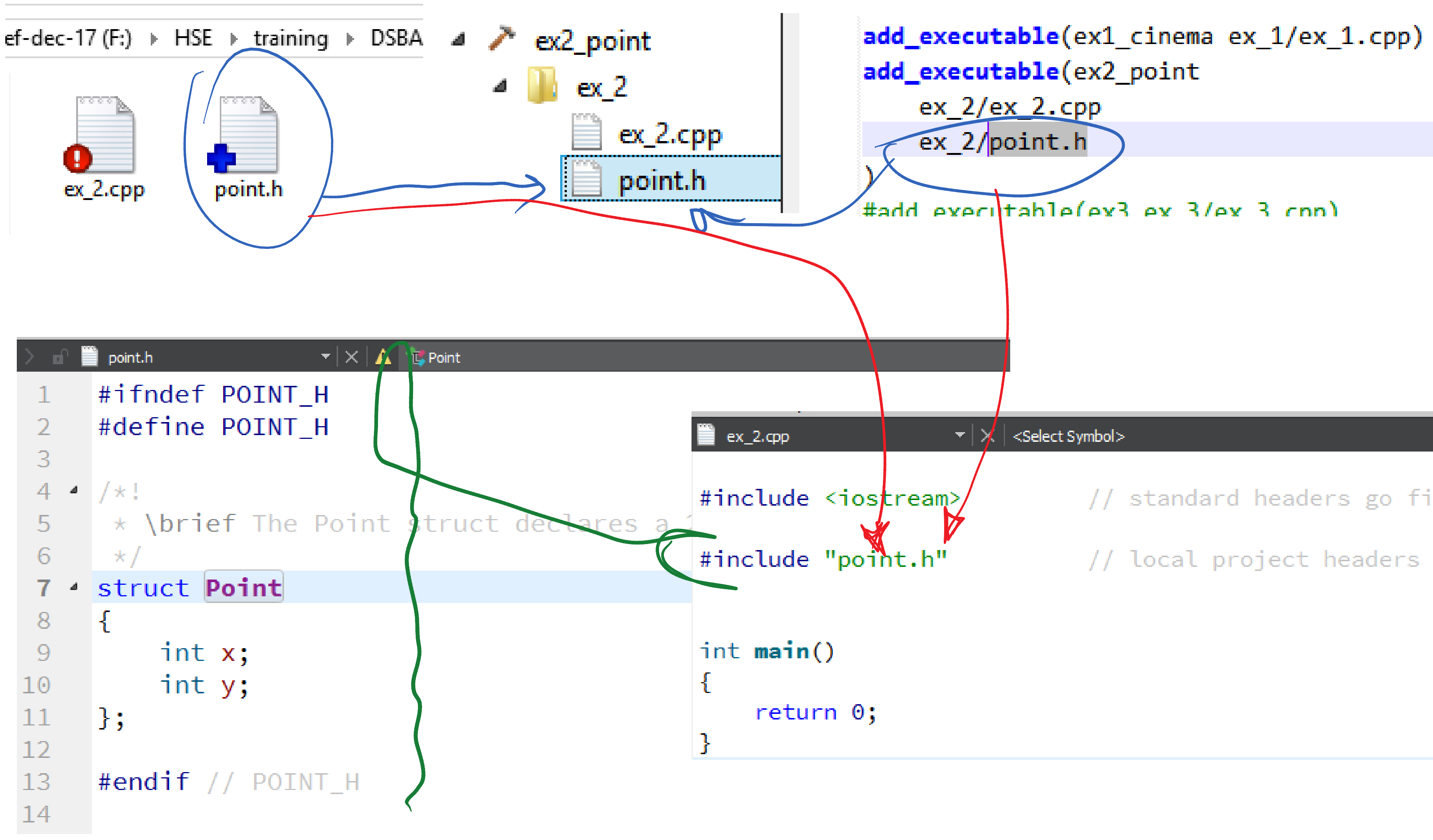
opening and closing braces

structure *members* (*fieds*)

terminates the structure declaration



Putting a Class Definition in a Header (.h) File



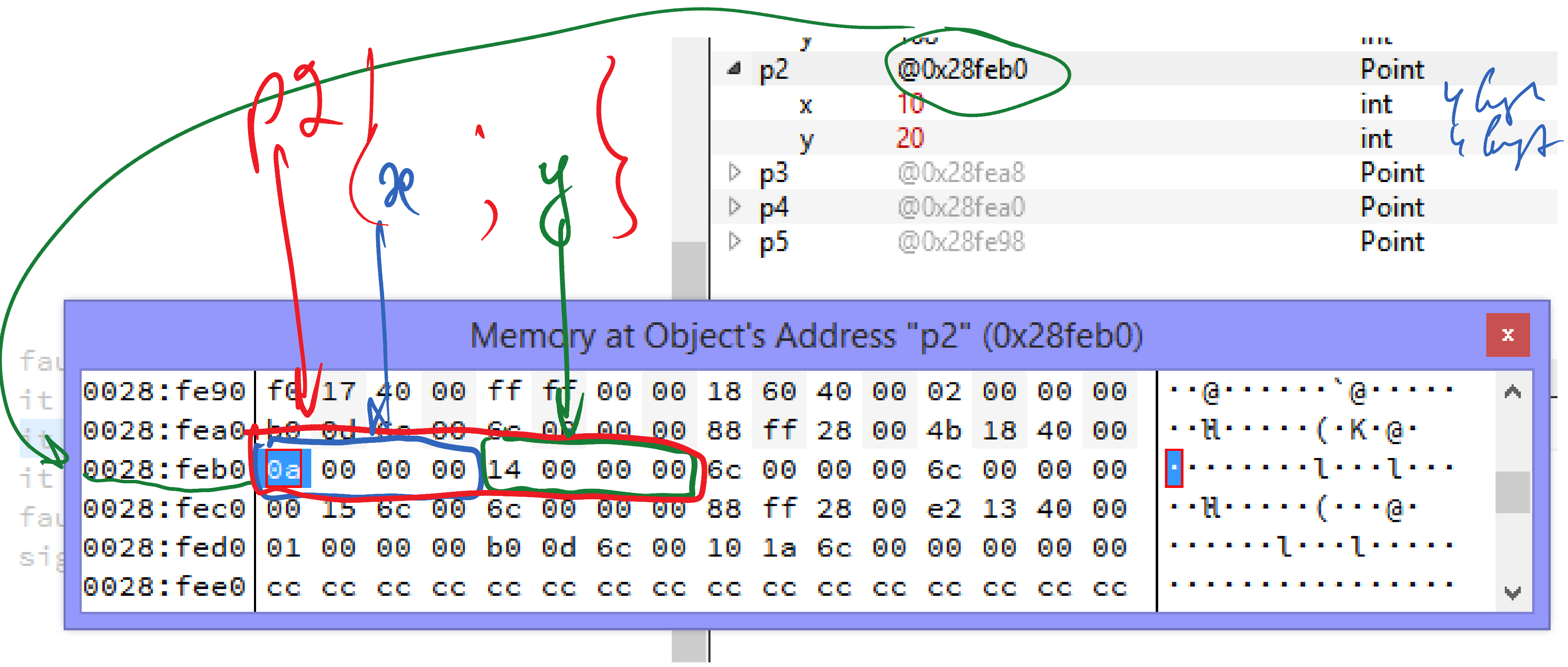
```
add_executable(ex1_cinema ex_1/ex_1.cpp)
add_executable(ex2_point
    ex_2/ex_2.cpp
    ex_2/point.h
)
#add_executable(ex3 ex_3/ex_3.cpp)
```

```
#ifndef POINT_H
#define POINT_H

/*!
 * \brief The Point struct declares a
 */
struct Point
{
    int x;
    int y;
};

#endif // POINT_H

```

```
#include <iostream>         // standard headers go fi

#include "point.h"          // local project headers


int main()
{
    return 0;
}
```

https://www.learncpp.com/cpp-tutorial/1-10a-header-guards/
https://stackoverflow.com/questions/4767068/header-guards-in-c

Representing `Point` Structure in Memory



◢ p2	@0x28feb0	Point
x	10	int
y	20	int
▷ p3	@0x28fea8	Point
▷ p4	@0x28fea0	Point
▷ p5	@0x28fe98	Point

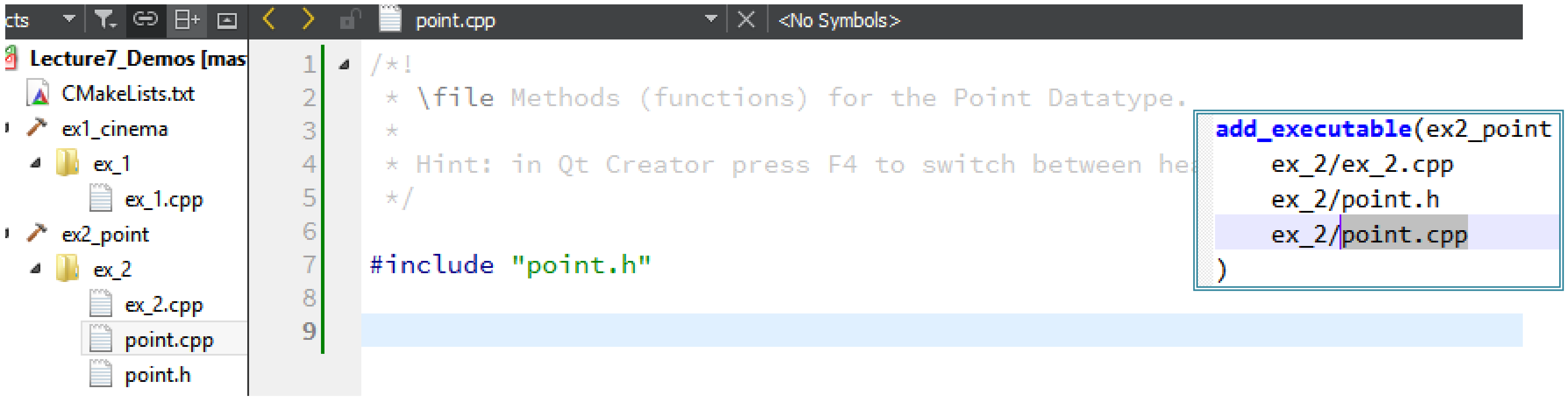
Memory at Object's Address "p2" (0x28feb0)

```
0028:fe90  f0 17 40 00 ff ff 00 00 18 60 40 00 02 00 00 00   ··@······`@·····
0028:fea0  b0 0d 6c 00 6c 00 00 00 88 ff 28 00 4b 18 40 00   ··l·····(·K·@·
0028:feb0  0a 00 00 00 14 00 00 00 6c 00 00 00 6c 00 00 00   ········l···l···
0028:fec0  00 15 6c 00 6c 00 00 00 88 ff 28 00 e2 13 40 00   ··l·····(···@·
0028:fed0  01 00 00 00 b0 0d 6c 00 10 1a 6c 00 00 00 00 00   ······l···l·····
0028:fee0  cc cc cc cc cc cc cc cc cc cc cc cc cc cc cc cc   ················
```

Add Some Operations on Points

- Adding and Subtracting two points
- Multiplication by a scalar
- Finding the length of a vector given by a point
  - Comparison of two points
- Sort a vector of Points (Contest 2 Problem 8)

Use a dedicated translation unit (point.cpp)!



```
/*!
 * \file Methods (functions) for the Point Datatype.
 *
 * Hint: in Qt Creator press F4 to switch between he
 */

#include "point.h"

```

```
add_executable(ex2_point
    ex_2/ex_2.cpp
    ex_2/point.h
    ex_2/point.cpp
)
```