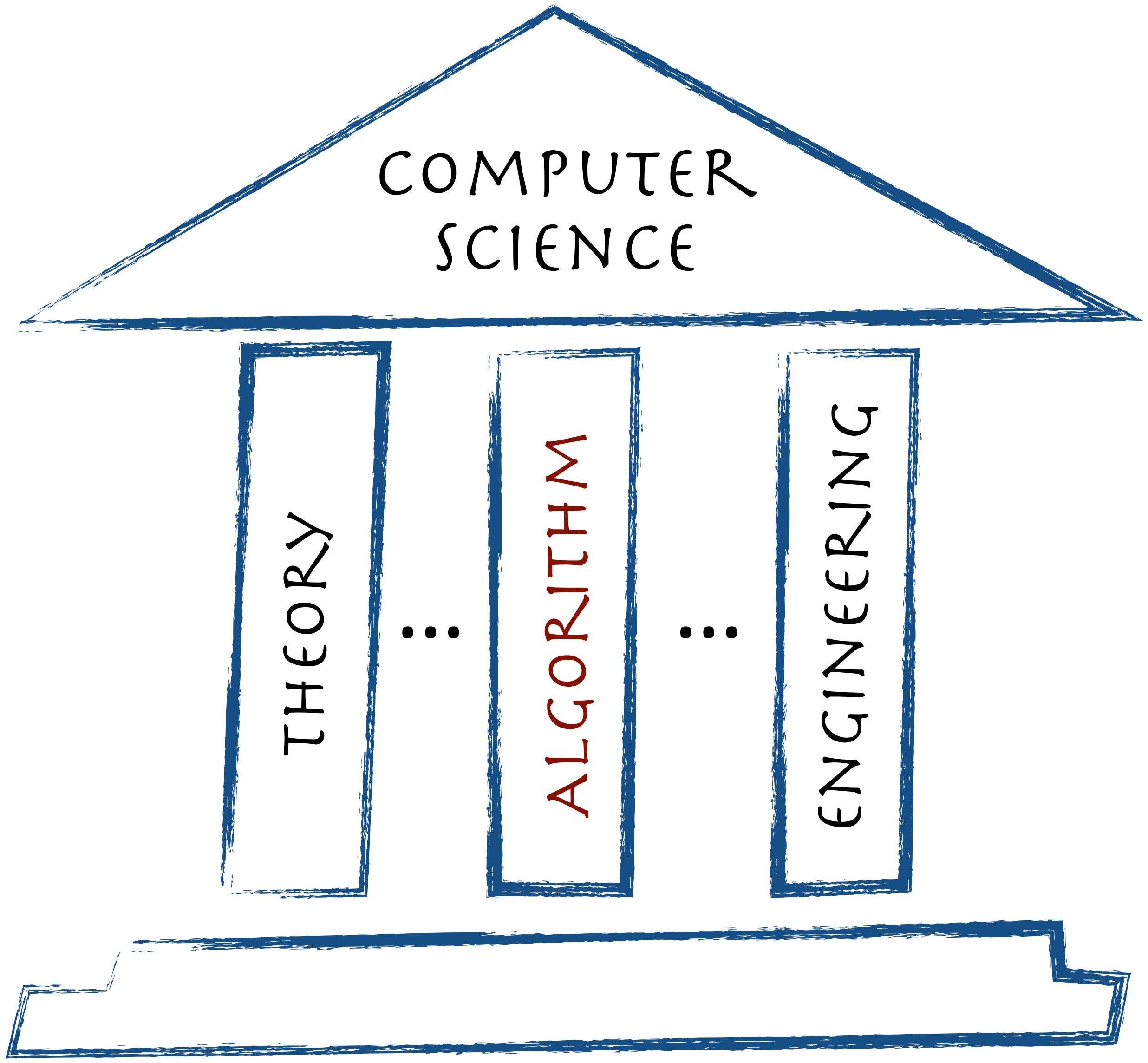
Algorithms Lab HS21

Angelika Steger, Emo Welzl
Ahmed Bouhoula, Giacomo Fabris, Hung Hoang,
Michael Hoffmann, Asier Mujika, Anouk Paradis, Kalina Petrova,
Meghana M. Reddy, Miloš Trujić, Manuel Wettstein, Xun Zou



COMPUTER SCIENCE

THEORY … ALGORITHM … ENGINEERING

A computer scientist is able to combine concepts and techniques from different areas in CS and apply them to solve problems in various application domains.

Applying Knowledge is difficult 🙁

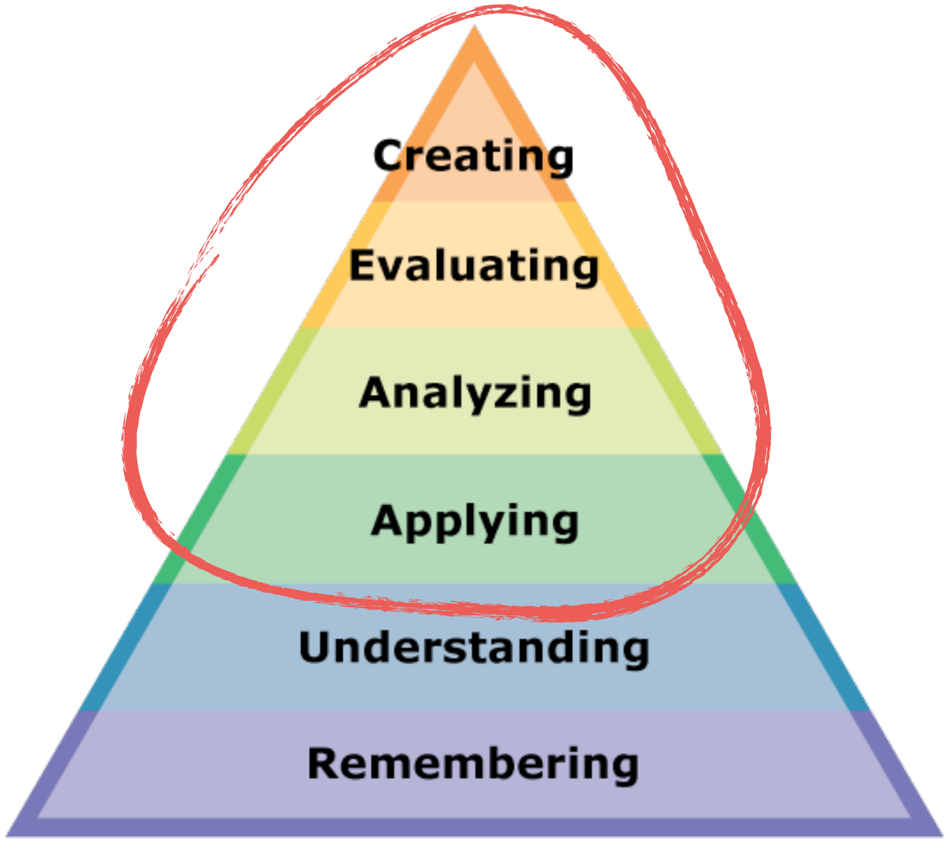
Typical coursework: Can you solve X using Y? (knowing it works)

Want: Do you know any way to solve X? (not knowing if possible)

Here: Can you solve X using a combination of $Y_1,\ldots,Y_k$?

Problem: There is no general algorithm to design algorithms.

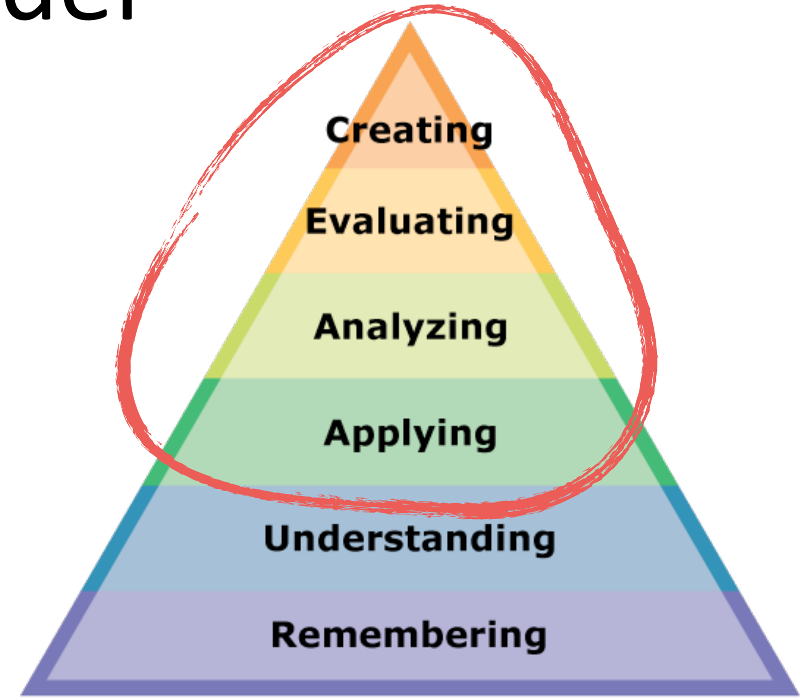
→ Practice over and over on concrete examples.



Taxonomy of learning objectives acc. to Bloom/Anderson
[Picture: http://www.learnnc.org/lp/pages/4719]

Goals

- develop intuition & experience in modeling and algorithm design: which techniques work for which problem?
- develop algorithm engineering skills: running a program on a real system vs. asymptotic analysis in an abstract machine model
- learn about some standard tools/ libraries for algorithms
- develop secondary skills: creativity, problem solving, time management, self-assessment, communication, testing,…



Taxonomy of learning objectives acc. to Bloom/Anderson

[Picture: http://www.learnnc.org/lp/pages/4719]

Objective: Master of Algorithms

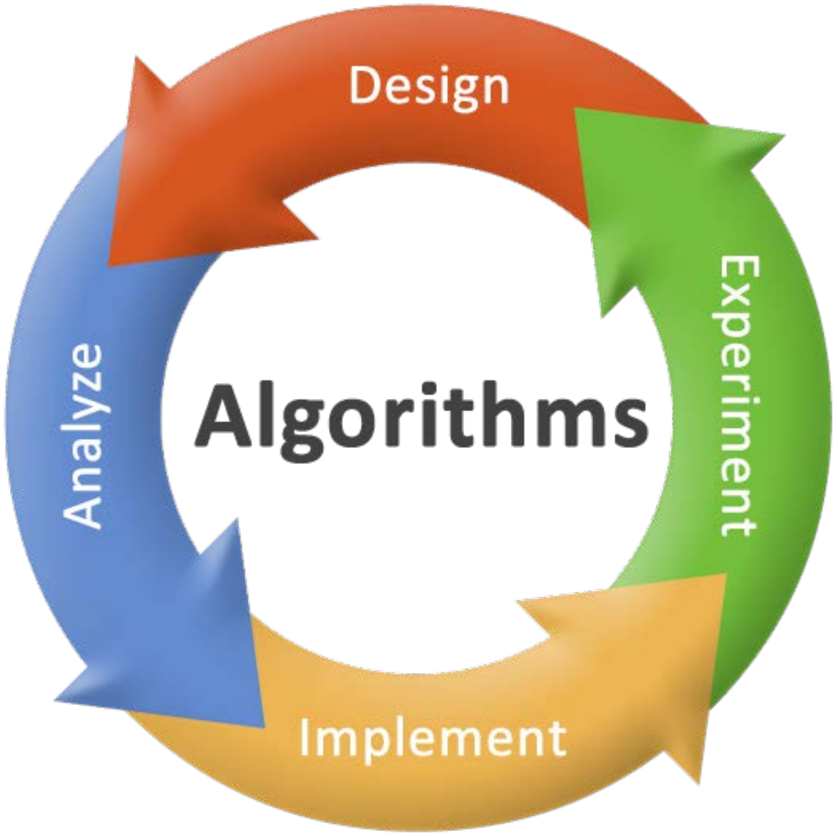
Design efficient algorithms for real-world problems.

In a toy world...

Problems are posed in form of a story.

Task:

- find an appropriate model
- design a suitable algorithm to solve it efficiently
- implement and test the algorithm on given data



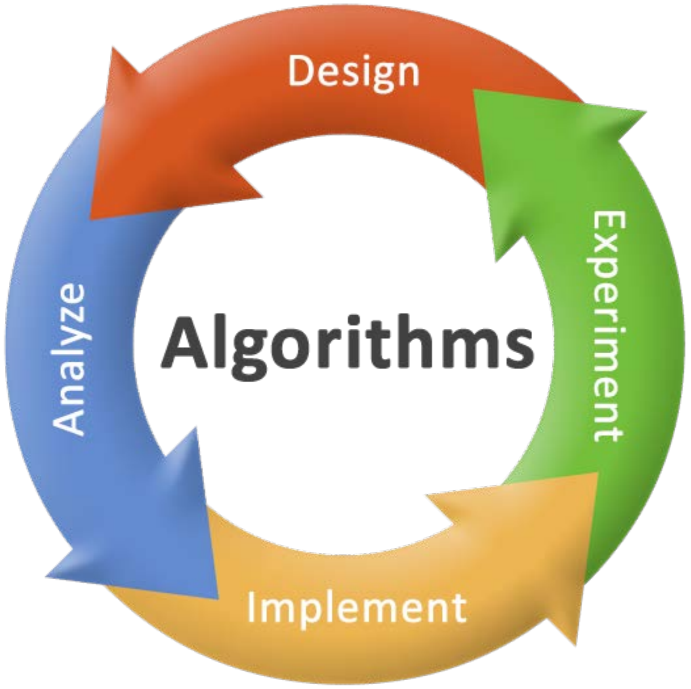
Time: 2h / problem

[Picture from http://openclassroom.stanford.edu/MainFolder/courses/IntroToAlgorithms/cs161logo.png]

Objective: Master of Algorithms

Usually the main challenge is here:

- find an appropriate model
- design a suitable algorithm to solve it efficiently

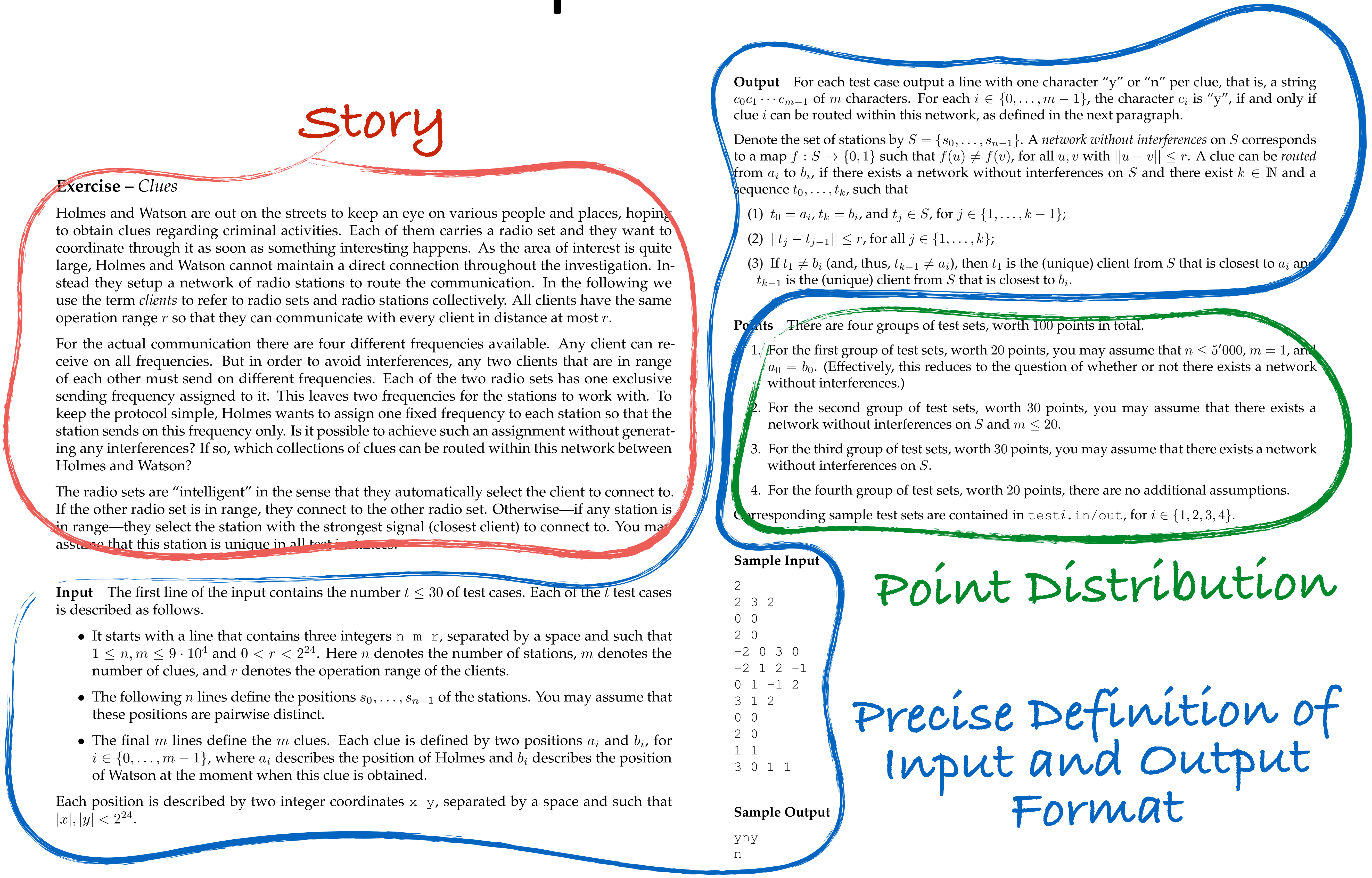


No need to reinvent the wheel => use libraries for standard data structures & algorithms.

All problems can be solved with ≤100 lines of (well formatted) C++ code.

[Picture from http://openclassroom.stanford.edu/MainFolder/courses/IntroToAlgorithms/cs161logo.png]

Example Problem



Story

Exercise – *Clues*

Holmes and Watson are out on the streets to keep an eye on various people and places, hoping to obtain clues regarding criminal activities. Each of them carries a radio set and they want to coordinate through it as soon as something interesting happens. As the area of interest is quite large, Holmes and Watson cannot maintain a direct connection throughout the investigation. Instead they setup a network of radio stations to route the communication. In the following we use the term *clients* to refer to radio sets and radio stations collectively. All clients have the same operation range r so that they can communicate with every client in distance at most r.

For the actual communication there are four different frequencies available. Any client can receive on all frequencies. But in order to avoid interferences, any two clients that are in range of each other must send on different frequencies. Each of the two radio sets has one exclusive sending frequency assigned to it. This leaves two frequencies for the stations to work with. To keep the protocol simple, Holmes wants to assign one fixed frequency to each station so that the station sends on this frequency only. Is it possible to achieve such an assignment without generating any interferences? If so, which collections of clues can be routed within this network between Holmes and Watson?

The radio sets are "intelligent" in the sense that they automatically select the client to connect to. If the other radio set is in range, they connect to the other radio set. Otherwise—if any station is in range—they select the station with the strongest signal (closest client) to connect to. You may assume that this station is unique in all test instances.

Input The first line of the input contains the number $t \leq 30$ of test cases. Each of the t test cases is described as follows.

- It starts with a line that contains three integers `n m r`, separated by a space and such that $1 \leq n, m \leq 9 \cdot 10^4$ and $0 < r < 2^{24}$. Here n denotes the number of stations, m denotes the number of clues, and r denotes the operation range of the clients.
- The following n lines define the positions $s_0, \ldots, s_{n-1}$ of the stations. You may assume that these positions are pairwise distinct.
- The final m lines define the m clues. Each clue is defined by two positions a_i and b_i, for $i \in \{0, \ldots, m-1\}$, where a_i describes the position of Holmes and b_i describes the position of Watson at the moment when this clue is obtained.

Each position is described by two integer coordinates `x y`, separated by a space and such that $|x|, |y| < 2^{24}$.

Output For each test case output a line with one character "y" or "n" per clue, that is, a string $c_0c_1 \cdots c_{m-1}$ of m characters. For each $i \in \{0, \ldots, m-1\}$, the character c_i is "y", if and only if clue i can be routed within this network, as defined in the next paragraph.

Denote the set of stations by $S = \{s_0, \ldots, s_{n-1}\}$. A *network without interferences* on S corresponds to a map $f : S \to \{0, 1\}$ such that $f(u) \neq f(v)$, for all u, v with $||u - v|| \leq r$. A clue can be *routed* from a_i to b_i, if there exists a network without interferences on S and there exist $k \in \mathbb{N}$ and a sequence $t_0, \ldots, t_k$, such that

(1) $t_0 = a_i$, $t_k = b_i$, and $t_j \in S$, for $j \in \{1, \ldots, k-1\}$;

(2) $||t_j - t_{j-1}|| \leq r$, for all $j \in \{1, \ldots, k\}$;

(3) If $t_1 \neq b_i$ (and, thus, $t_{k-1} \neq a_i$), then t_1 is the (unique) client from S that is closest to a_i and t_{k-1} is the (unique) client from S that is closest to b_i.

Points There are four groups of test sets, worth 100 points in total.

1. For the first group of test sets, worth 20 points, you may assume that $n \leq 5'000$, $m = 1$, and $a_0 = b_0$. (Effectively, this reduces to the question of whether or not there exists a network without interferences.)
2. For the second group of test sets, worth 30 points, you may assume that there exists a network without interferences on S and $m \leq 20$.
3. For the third group of test sets, worth 30 points, you may assume that there exists a network without interferences on S.
4. For the fourth group of test sets, worth 20 points, there are no additional assumptions.

Corresponding sample test sets are contained in `testi.in/out`, for $i \in \{1, 2, 3, 4\}$.

Sample Input

```
2
2 3 2
0 0
2 0
-2 0 3 0
-2 1 2 -1
0 1 -1 2
3 1 2
0 0
2 0
1 1
3 0 1 1
```

Sample Output

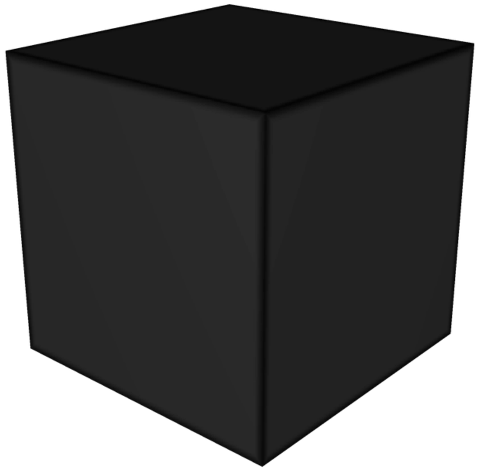
```
yny
n
```

Point Distribution

Precise Definition of Input and Output Format

Black Box Evaluation

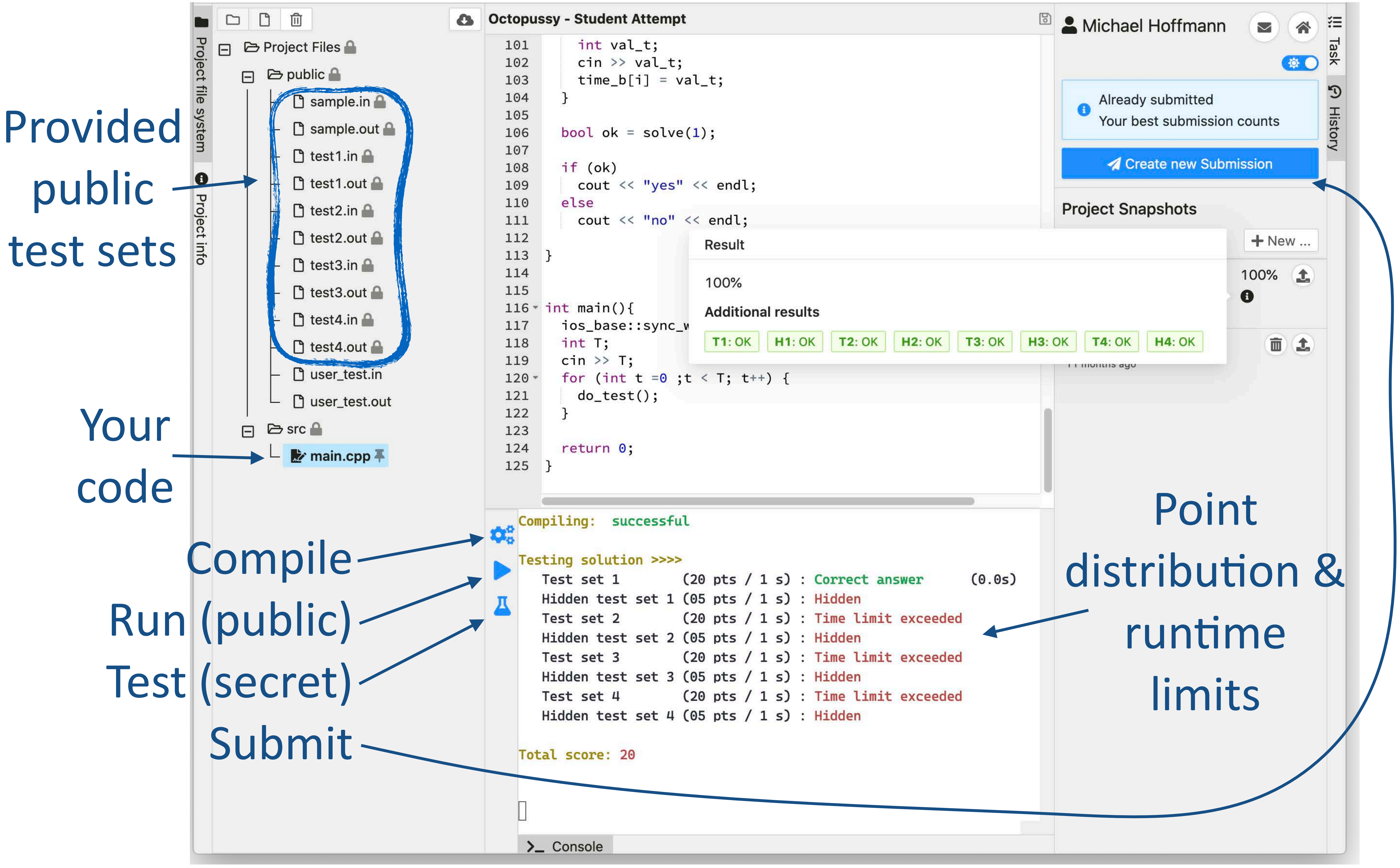
```
string sInput;
int iLength, iN;
double dblTemp;
bool again = true;

while (again) {
    iN = -1;
    again = false;
    getline(cin, sInput);
    system("cls");
    stringstream(sInput) >> dblTemp;
    iLength = sInput.length();
    if (iLength < 4) {
        again = true;
        continue;
    } else if (sInput[iLength - 3] != '.') {
        again = true;
        continue;
    } while (++iN < iLength) {
        if (isdigit(sInput[iN])) {
            continue;
        } else if (iN == (iLength - 3) ) {
```



CORRECT
WRONG ANSWER
TIMELIMIT
SEGMENTATION FAULT
ASSERTION FAILURE

- automatic evaluation by online judge
- 4-5 groups of 20-30 test instances each
- given timeframe (X sec. to complete)
- only completely solved groups score
- arbitrary many submissions, the best is graded

Online Judge: Code Expert



Provided public test sets

Your code

Compile

Run (public)

Test (secret)

Submit

Point distribution & runtime limits

Already submitted
Your best submission counts

Result

100%

Additional results

T1: OK H1: OK T2: OK H2: OK T3: OK H3: OK T4: OK H4: OK

```
    int val_t;
    cin >> val_t;
    time_b[i] = val_t;
  }

  bool ok = solve(1);

  if (ok)
    cout << "yes" << endl;
  else
    cout << "no" << endl;

}


int main(){
  ios_base::sync_w
  int T;
  cin >> T;
  for (int t =0 ;t < T; t++) {
    do_test();
  }

  return 0;
}
```

```
Compiling:  successful

Testing solution >>>>
   Test set 1         (20 pts / 1 s) : Correct answer      (0.0s)
   Hidden test set 1  (05 pts / 1 s) : Hidden
   Test set 2         (20 pts / 1 s) : Time limit exceeded
   Hidden test set 2  (05 pts / 1 s) : Hidden
   Test set 3         (20 pts / 1 s) : Time limit exceeded
   Hidden test set 3  (05 pts / 1 s) : Hidden
   Test set 4         (20 pts / 1 s) : Time limit exceeded
   Hidden test set 4  (05 pts / 1 s) : Hidden

Total score: 20
```

Environment & Tools

On Code Expert:

- GNU C++ compiler V8.4.1, using C++14
- … and the C++ std library that comes with it
- Boost and the Boost Graph Library (BGL) V1.77
- Computational Geometry Algorithms Library (CGAL) V5.2.3

Will be introduced in this course.

Previous knowledge & self-study.

Documentation at algolab.inf.ethz.ch/doc.

Getting Started with C++

- @Everyone: Read over the document "A Short Introduction to C++ for the Algorithms Lab" (even if you are a C++ expert)!
- We also provide a virtual box image that you can use to develop on your own computer, e.g. offline (s.a. the explanation video on moodle).

Next:

- Sample problem: Even Pairs
  - Four solutions and
  - all you need to know to solve problems & submit solutions to CodeExpert.

Then:

- Course Organization

Prerequisites

www.cadmo.ethz.ch/education/lectures/HS21/algolab/prereqs

- Strategies: Brute force, greedy, divide & conquer, dynamic programming, backtracking, binary search
- Data structures: Array, stack, set, queue, search trees, heap, hash table, union-find
- Graph algorithms: DFS, BFS, MST, Dijkstra
- Graph concepts: Directed graph, coloring, matching, topological sorting, (strongly) connected components, matchings

Occasionally we do a recap, but *not* full explanations.

Course Outline

we are here →

1	FUNDAMENTAL ALGORITHMS
2	
3	
4	
5	ADVANCED ALGORITHMS
6	
7	
8	
9	
10	EXAM PREPARATION
11	
12	
13	
14	no tutorial on Dec 22

Course Format

- Tutorial: Wednesday 16–18
  - background and technical issues
  - short recap of known algorithmic concepts with examples (also teach a few new ones, but focus on applications rather than theory)
  - 3-fold problems: individual work
- Problem of the week: Monday 14–16
  Exam like conditions: one problem, 2h to solve
- Individual Work: ~12h / week
- Consulting: Friday 16–18 (until Oct 15)

  Optional
- Assessments: Tue, Thu, Fri 16–18 (from Oct 19)

Course Workload

Tutorial:	2 h × 13 =	26 h
Problem of the week:	2 h × 13 =	26 h
Individual Work:	12 h × 13 =	156 h
Exam preparation:		22 h
Exam:		10 h
Total:		**240h**

= 8 ECTS credits (of 30h each)

- Every Wednesday: new set of problems (3–5)
- Students submit their solutions within one week.
- Automated grading/feedback by an online judge
- We provide solutions for a few selected problems.

Important: Take the time and make the effort to work out the solutions to these problems by yourself.

Picture: [https://pixabay.com/en/board-font-problem-solution-chalk-1521347/]

Point Distribution

- Make sure to carefully read the “Points” section in the problem description!
- Partial solutions are possible and (sometimes considerably) easier to get than a full solution.

Rules of thumb:

- Exam problems have >= 3 different levels.
- Everybody should be able to get >= 50 points with some work & preparation.
- The last <= 20 points may require an additional idea (can be a challenge in some problems).

Moodle

moodle-app2.let.ethz.ch/login

use NETHZ account / NETHZ password

There you can

- download slides of the tutorials,
- download problem sheets, and
- discuss problems with your colleagues in the forums.

Moodle is the primary communication platform for this course => Make sure to read message boards.

Code Expert

https://expert.ethz.ch/enroll/AS21/algolab

use NETHZ account / NETHZ password

There you can

- test your solutions,
- submit your solutions, and
- see the actual timelimits for the testsets.

XP are awarded for submissions within two weeks for regular problems and within two hours for PotW. (These have no impact on the grade.)

Problem of the Week (PotW)

- Exam-like problem posted every Monday at 14:00
- Solve it within the next 2 hours.
- Use this opportunity to assess your skills. To get a realistic assessment, only use resources also available during the exam (→ `algolab.inf.ethz.ch/doc`).
- (PotW and exam only) Some testsets are hidden, you do not see the results during the 2/5 hours period.
- (PotW and exam only) Questions about the problem statement must be submitted as a message on Code Expert.

Exam

The grade is based solely on the exam.

- 6 problems in 10h = 2 x 5h
- Two known* problems (one on each exam day)
- HG computer rooms, no custom hardware
- Submission/judging exactly as during the semester
- Documentation on `algolab.inf.ethz.ch/doc`
- No additional material
- Repetition: retake course in HS22

* The algorithmic problem is exactly the same as in a problem that is posed during the course; the story is different, though.

Exam Computers

Based on the Fedora setup in the HG student labs (e.g., HG E26.1/3) ==> Check it out!

- Most standard tools that you expect on a Linux system should be there.
- If you find something to be missing: Report as early as possible on the Exam Tools forum on moodle.
- (No guarantees, but if there is a Fedora package with standard licensing it may be included…)
- ch-de keyboard layout

Individual Performance Assessments

- 1-on-1 discussion with an assistant (<15min)
- about specific topic/problem from the week(s) before
- topic known in advance => prepare
- Goals:
  - reflect about problem and your solution
  - practice oral communication (interview)
  - get feedback on where you stand
- How? individual appointment via electronic system

Individual Performance Assessments

- 3 Windows of 3 weeks each over the semester: every student gets 1 appointment per window
- When? Tue, Thu, Fri 16–18
- Every student who completes all three assessments receives a bonus in form of a quarter-grade (+0.25) on their grade from the final exam.

Individual Performance Assessments

we are here →

1	
2	
3	
4	
5	Assessment Window #1 Oct 19 – Nov 5
6	
7	
8	Assessment Window #2 Nov 9 – Nov 26
9	
10	
11	Assessment Window #3 Nov 30 – Dec 17
12	
13	
14	no tutorial on Dec 22

Individual Appointments

New Integrated in Code Expert.

- Appointments are for a 1h slot (16–17 or 17–18)
- Registration opens one week before the window starts (e.g., Tue, Oct 12, 16:00 for Window #1)
- You will be called (into the room or by receiving a Zoom link).
- Prepare to present your solution / ideas.
- Ternary Feedback, from 0 🙁 to 2 😀 both ways
- Assessment completed by giving feedback to the assistant in the electronic system.

Getting Help

- Your best bet for quick help are our forums, where other students or an assistant can help you out;
- or use the consulting hours.
- Important: Use hints with caution; there are no forums in the exam...

For administrative or technical problems (e.g., with CodeExpert or moodle) use

algolab@lists.inf.ethz.ch

All other questions should be discussed on the forums.

———————— That’s it for today. ————————