FRESNO STATE

Discovery. Diversity. Distinction.

Algorithms and Data Structures (CSci 115)

California State University Fresno

College of Science and Mathematics

Department of Computer Science

H. Cecotti

Outline

- Introduction
- Syllabus
- Tests & Assignments
- Conclusion

Warning

- A large number of elements that you will implement in this course exist already in standard libraries in many languages (class vector, sort algorithm)
  - Because it is very useful
- **Typical issues**:
  - You may feel that you reinvent the wheel
  - You may dislike the course due to the theoretical part
- But:
  - **Job interviews are based on exercises related to the content of this course**
- As students
  - You must understand perfectly what is happening under the hood
  - You must be able to do what exists already on your own
  - A large number of problems require a fine knowledge of what you will see in the classes and labs
- → You must enter this course with the right state of mind

Algorithms and Data structures

- Motivations
  - A key course in computer science.
    - Very important and directly related to many courses
  - Data structures: used everywhere
    - Operating systems (queues, stacks, trees, Red & Black tree in linux kernel…)
    - Databases (hash table, B-trees,….)
    - Machine learning (graphs, trees, matrices, tensors,…)
  - **Everything** that you will see in this course has real applications
- Combination
  - Programming (C++) and Theoretical analysis
- Qualities to be developed
  - To think outside of the box
  - Rigorous
  - Resilient (to bugs, need of debugging)
  - What if…
    - To think about all the different possibilities

Algorithms and Data structures

- Programming
  - Save and keep your files very well **organized**
    - With comments
    - With meaningful names for the variables
  - /!\ Code must be easy to read and understand
    - You come back next year in your code: you must understand what it is about
      - Why: because you will need this code for other projects
- Data structures → Creation of templates to be reused in multiple applications
  - A way of organizing and storing data in a computer
    - → to be **accessed** and **modified** efficiently

Algorithms and Data Structures

- Managing information/data
  - What to do with it?
    - To **create** information
      - By storing the information in a judicious structure
    - To **access** the information (search information)
      - Based on its content
      - Based on a key relative to its content
    - To **organize** data
      - To delete
      - To add (to insert)
      - To sort

Algorithms and Data Structures

- Common functions to implement in a data structure
  - **C**reate, **R**ead, **U**pdate, and **D**elete (CRUD)
- Common functions for multiple data structures
  - Object Oriented Programming
    - Focus on the structure of the different classes
      - 1 or more class(es) / data structure
    - Hierarchy + Abstract Classes

Syllabus (1)

- Schedule
  - When
    - Monday (class): 9h00-9h50 AM
    - Wednesday (class): 9h00-9h50 AM
    - Friday (class): 9h00-9h50 AM
  - Where
    - **Industrial Tech Bldg Rm 294**
- Labs
  - See respective schedule depending on the group

Syllabus (2)

- Contact:
  - Dr Hubert Cecotti (classes)
  - Dr Dhanyu Amarasinghe (labs)
  - Office hours:
    - Tuesday-Thursday 9-11
  - Email:
    - In the object, you **must** put the string “[CSCI115]”.
    - Be formal in the content of the email, it is not for instant messages.
      - For a discussion, for questions that require long exchanges, it is better to come directly to the office.
- Do not wait that it is too late to contact me!
  - Feedback and comments are very welcome

Syllabus (3)

- Canvas
  - All the material from the different classes will be available on Canvas
  - Class + Labs
  - If you print the pdf presentation to the classes
    - **DO** take notes!
    - The slides are **NOT** enough for succeeding!
    - Try and test all the different elements
      - In small examples
      - In a personal project that you can include as part of your portfolio for finding a job

Syllabus (4)

- Software/technologies that will be used:
  - Programming language:
    - C++
  - Windows
    - Visual Studio Community
    - GCC …
- Urgent
  - Get a C++ compiler on your machine

Syllabus (5)

- Midterms
  - 2 midterms
    - On Canvas
      - Definitions
      - Multiple choices
      - Code to write
      - Code tracing
    - Focus on both C++ programing and theoretical analysis
  - Dates and time
    - You can find the provisional dates of the midterms in the syllabus document
    - The time and dates for the Midterms and the Project may change
    - The information will be given in class and on Canvas

Materials (1)

- Additional contents
  - Links to video, documents, will be given on Canvas
    - Some algorithms are graphically represented on youtube (e.g. sorting algorithm)
  - There are many books in open access, available at the library
    - You can check it online ““name of what you want” book pdf” on google.
    - Pdf files: books are large files, no need to print! Just use it as a reference.
  - To be able to try everything on your computer!
    - In this course, **practice** is very important.
      - Without the implementation, it can be hard to catch some subtle concepts
    - You must program the different algorithms: learning while doing !
- The best way to understand
  - To **finish** the labs
  - To implement all the different elements
    - Without looking at solutions online

Materials (2)

- Lots of documents, tutorials, video online
  - ➢ **BUT**: not very well organized: easy to get lost, discouraged ☹
  - ➢ Be careful to blogs written by random people!
- Some classic recommended links and books:
  1. Introduction to Algorithms, 3rd Edition, by T.H. Cormen, C.E. Leiserson, R.L. Rivest, C. Stein, The MIT Press.
  2. Algorithms and data structures, by N. Wirth (available online for free)
  3. Data structures and Algorithm analysis in C++, 4th Edition, by M. A. Weiss, Pearson.
- Just to read = useless → Read **AND** Implement
  - ➢ Just be reading, it is hard to grasp all the details
  - ➢ It is hard to know you understand until you implement yourself
    - o Easy to miss important details by just reading

Learning outcomes

- A solid understanding of fundamental data structures
- A solid skill of problem solving in programming
  - by choosing the appropriate data structures for a given problem
- Programming in C++
  - Object oriented programming for better organizing the code and data structures
  - Full control of memory management
- Key understanding about the advantages and constraints of different data structures.
  - E.g. Impact of multi-threading
- Team spirit to solve larger scale problem and use current social media tools to communicate efficiently and share files
  - To deal with deadlines
- Presentation of your software effectively
  - write well-structured and well-presented reports
  - to communicate how their applications can be used with both computer science professionals and general audience.

Learning outcomes

- To succeed this course, you need:
  - To be **rigorous**
    - **Miderm & Final: no partial credit for typo or related errors**
  - To properly organize your files (labs, classes,…)
  - To be present during classes and in labs

Tests & assignments (1)

- Marking
  - **Attendance and participation**: 5%
  - **Lab assignments**: 15%
    - what you need to finish for the next week
  - **Project**: 20%
    - Turn-based video game
      - In a maze, avoid bad guys, find the treasure…
  - **Midterm 1**: 15% (from week 1 to the week of Midterm 1)
    - Canvas + Lockedown browser
  - **Midterm 2**: 15% (from the week of Midterm 1 to the week of Midterm 2)
    - Canvas + Lockedown browser
  - **Final exam**: 30% (comprehensive: from week 1 to week 16)
    - Canvas + Lockedown browser

- A: [85-100]%
- B: [70-84]%
- C: [55-69]%
- D:[40-54]%
- F: [0-39]%

Provisional schedule

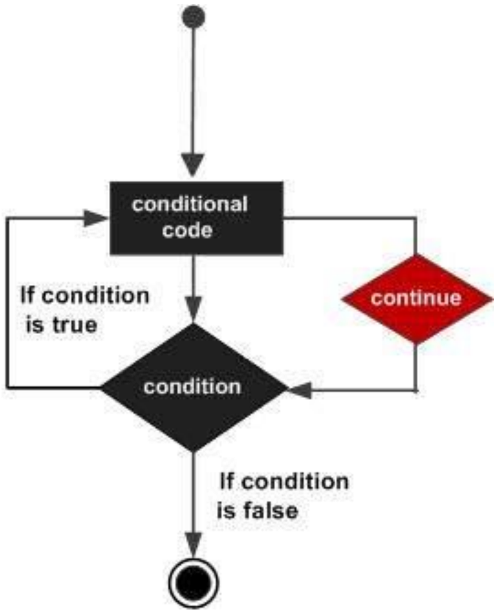
- Given on Canvas
  - …

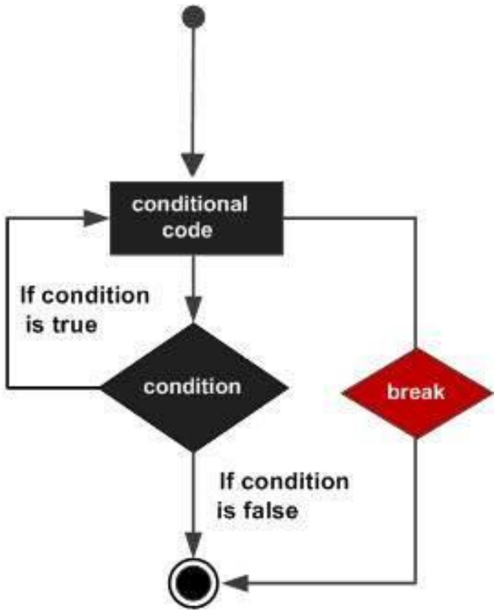
Lab …

- All about implementations…
  - Prepare the labs…
    - Paper + Pen : from the theory to the pseudo-code
    - → Implementation
      - No waste of time you know what you are doing
  - **Finish** the labs…
- Pitfalls
  - Trial and errors without understanding the rules
    - It may work at the end but you remain insecure with what you are doing
    - → Waste of time, problems with understanding the theory
  - Issues with memory managements
    - New / Delete
  - Issues with inputs/outputs in functions
    - Pointers

Labs and projects

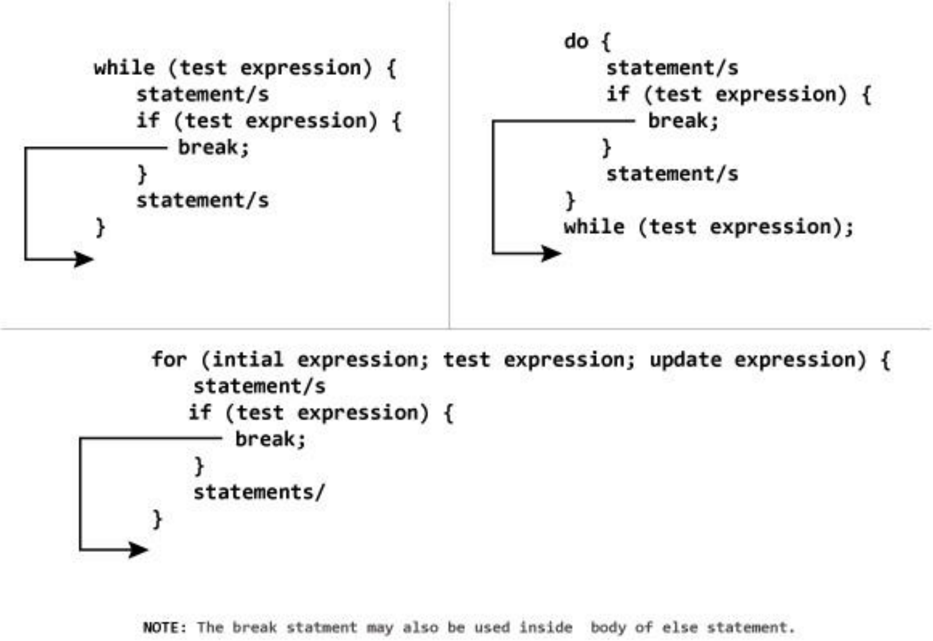
- C++
  - No **break** or **continue** in the code to exit a loop!





Labs and projects

- C++
  - ➢ No **break** or **continue** in the code to exit a loop.
- What I want: clear well written commented code with **functions**

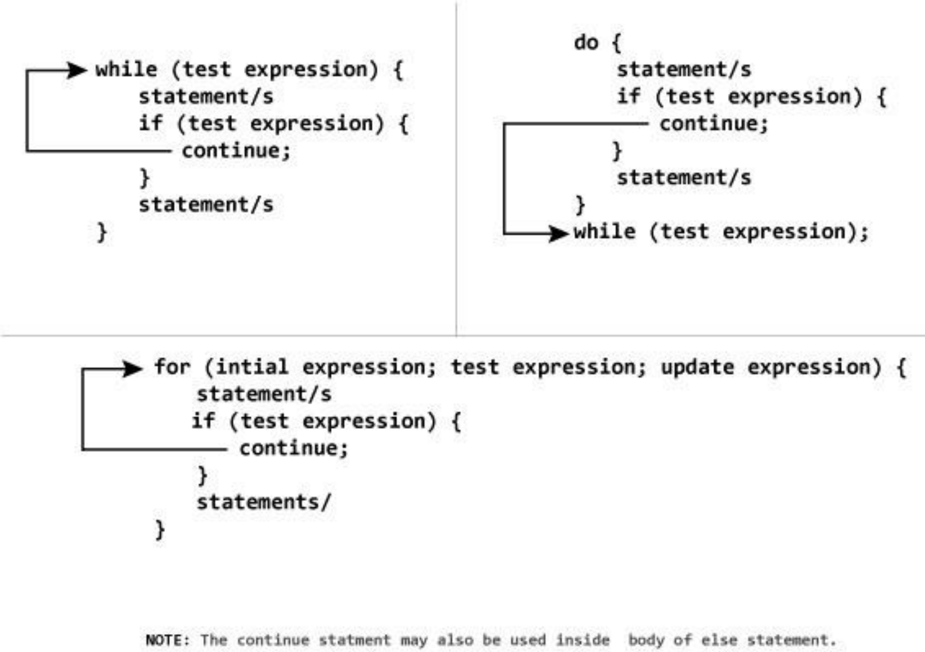


```
while (test expression) {
    statement/s
    if (test expression) {
        break;
    }
    statement/s
}
```

```
do {
    statement/s
    if (test expression) {
        break;
    }
    statement/s
}
while (test expression);
```

```
for (intial expression; test expression; update expression) {
    statement/s
    if (test expression) {
        break;
    }
    statements/
}
```

NOTE: The break statment may also be used inside body of else statement.



```
while (test expression) {
    statement/s
    if (test expression) {
        continue;
    }
    statement/s
}
```

```
do {
    statement/s
    if (test expression) {
        continue;
    }
    statement/s
}
while (test expression);
```

```
for (intial expression; test expression; update expression) {
    statement/s
    if (test expression) {
        continue;
    }
    statements/
}
```

NOTE: The continue statment may also be used inside body of else statement.

Exercise

- Warm up
  - Create a function that takes an array A as an input and return a Boolean if there exists a value in the array superior to x, which is given as an input.
    - bool f(int* a, int x);
      - With a for loop, using the break statement
      - With a while loop
      - With a do-while loop
      - With a if condition, with the ternary conditional operator
        - (condition) ? (if_true) : (if_false)

Questions ?

- Attendance on Canvas
- Remark
  - ➢The instructor is not a compiler.
  - ➢The instructor will not look at syntax error
  - ➢The instructor will not debug your labs and projects