Python programming

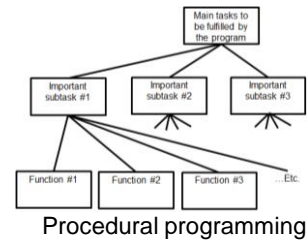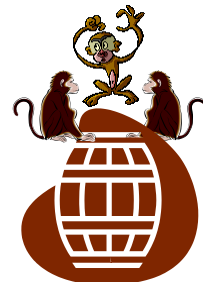Procedural programming

Introduction To CPSC 231

James Tam

Problem solving

Fun! Fun! Fun!

Administrative (James Tam)

- Contact Information
  - Office: ICT 707
  - Email: tam@ucalgary.ca
  - Make sure you specify the course name and number in the subject line of the email 'CPSC 231'
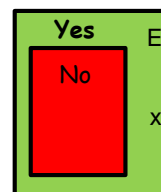
- Office hours
  - Office hours: Tue & Thur 16:00 – 16:30
  - Appointments possible at other days/times (subject to availability).
  - Dropping by outside of office hours without prior notice is "hit and miss"

My Office

ICT 7th

Yes

E

No

x

Course Resources

- Required resources:
  - Course website:
    http://pages.cpsc.ucalgary.ca/~tamj/2017/231P/index.html (You **must** get the notes off the course webpage before lecture)
- Recommended but not required:
  - "*Starting Out with Python*"(Gaddis T.) Addison-Wesley.
  - Alternatively you can access any book licensed by the university ('for free") on the library web site:
  - (One of many books available) "*Visual QuickStart guide*"
    http://proquest.safaribooksonline.com.ezproxy.lib.ucalgary.ca/

How To Use The Course Resources

- They are provided to support and supplement this class.
  - The notes outline the topics to be covered
  - *At a minimum* look through the notes to see the important topics.
  - However the notes are just an outline and just looking at them without coming to class isn't sufficient to do well
  - You will get the details (e.g., explanations) during lecture time
    - Take notes!

Your Engagement Level -±-> Your Learning

Learning

Computer
programs

Computer
program

James Tam
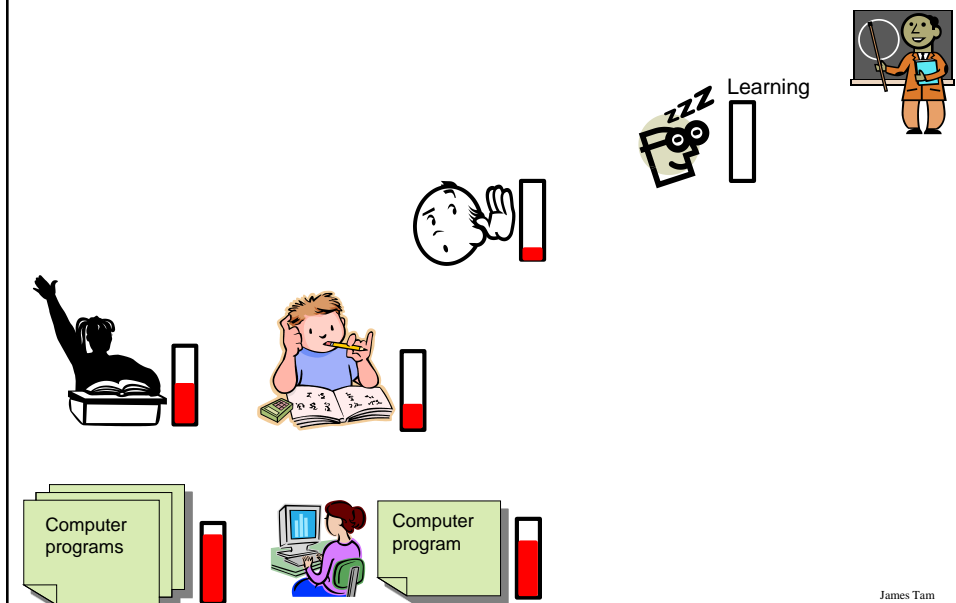
How To Use The Course Resources (2)

```
def display (world):

    sys.stdout.write(' ')
    for i in range (0, columns, 1):
        if (i < 10):
            print i,
        else:
            num = i + 55
            ch = chr(num)
            print ch,
    print

    for i in range (0, columns, 1):
        sys.stdout.write(' -')
    print

    for r in range (0, rows, 1):
        for c in range (0, columns, 1):
            sys.stdout.write('|')
            sys.stdout.write(world[r][c].appearance)
        print ('|'), r
        for i in range (0, columns, 1):
            sys.stdout.write(' -')
        print
```

James Tam

How To Use The Course Resources (2)

```
def display (world):

            write(' ')
            ge (0, columns, 1):
                0):
                    print i,

                    num = i - 55
                    ch = chr(num)
                    print ch,

        for i in range (0, columns, 1):
            sys.stdout.write(' -')
        print

        for r in range (0, rows, 1):
            for c in range (0, columns, 1):
                sys.stdout.write('|')
                sys.stdout.write(world[r][c].appearance)
            print ('|'), r
            for i in range (0, columns, 1):
                sys.stdout.write(' -')
            print
```

When you are in class make sure that you supplement the slides with your own notes (because you aren't going to remember it in the exams if you don't)

If you miss a class make sure that you catch up on what you missed (get someone's class notes)

James Tam

How To Use The Course Resources (3)

- •What you are responsible for:
  - Keeping up with the content in class which includes the topics covered but also announcements or assignment information whether you were present in the class or not.
  - If you are absent, then you are responsible for getting the information from the other students in class.
    - •Make sure your UC registered email is correct and one you actually read (some course announcements will be sent to these emails)
  - (I won't be able to repeat the lecture content if you are absent…there's just too many of you to make it practical and recall to get the most out of the class you need to be actively engaged)

- •However, after you've caught up by talking with a classmate:
  - Ask for help if you need it
  - There are no dumb questions
  - …except for waiting until the exam
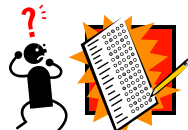
James Tam

Tam's "House Rules"

•I will endeavor to keep the lecture within the prescribed time boundaries

•You won't pack up and end before time is up

Tam's "House Rules"

•No recordings/captures without permission during class please

•(Recall that learning tends to increase with additional levels of engagement).

Yes: "This Stuff Will Be On The Exam"

- The administrative notes contains important information e.g. how your grades are calculated, course policies etc.
- To encourage students to pay attention to details (and to reward those who do so):
  - Some of your midterm multiple questions will come from this section.
  - You may see a question or two from this section on the final exam as well.

Evaluation Components

- Four mini assignments (*2/100 of term grade*)
- Five full assignments (*33/100 of term grade*)
- Two examinations (*65/100 of term grade*)

Assignments

- You will create a working and executable computer program.

- Use a text editor (similar to a word processor minus the fancy formatting capabilities) to create it and you will electronically submit the text file (to D2L) for marking.

- Although you may be given some time in tutorial to work on your assignments (during the "open tutorial") mostly you will complete your work on your own time.
  - Don't underestimate the time/effort required.
  - Creating a good working program is harder than it may first appear.

- Assignments will be marked by the tutorial instructor.
  - He/she is the first person to go to if you want to determine your grade or have questions about grading.

Assignments (2)

- Collaboration:
  - Each student must work on his/her own assignment (no group work is allowed for this class)
  - Each student must individually submit an assignment
  - Students must not see each other's assignment code
  - Additional details will be provided later during the semester

- You will electronically submit the file which contains your solution to the assignment via D2L:
  - http://d2l.ucalgary.ca/
  - (Find the appropriate course name/number and lecture section)

Submitting Assignments

- **Bottom line: it is each student's responsibility to make sure that the correct version of the program was submitted on time.**
  - Alternate submission mechanisms e.g., email, uploads to cloud-based systems such as Google drive, time-stamps, TA memories cannot be used as alternatives if you have not properly submitted into D2L
  - **Only files submitted into D2L by the due date is what will be marked**
- Late assignments will not be accepted.
- If you are ill then medical documentation is required.
  - Contact your **course instructor** and not your tutorial instructor to get permission for a late submission

JT's Helpful Hint: Electronically Submitting Work

- Bad things sometimes happen!
  - A virus, hardware failure, you screwed up the submission.
- Rules of thumb for assignment submissions:
  - Do it early! (Get familiar with the system)
  - Do it often! (If somehow real disaster strikes and you lose everything at least you will have a partially completed version that your TA can mark).
  - *Check your work*.
    - Don't assume that everything was submitted OK.
    - Don't just check file names but at least take a look at the actual file contents (not only to check that the file wasn't corrupted but also that you submitted the correct version).
    - Assignment 0 'A0':
      - An exercise in tutorial where you practice submitting and checking your work
      - Not directly graded but still important to complete

How To Verify Submissions In DropBox

- There is a help link provided with each assignment description.
- Teaching Assistants will cover in conjunction with Assignment zero.
  - Not graded but important practice
- Resource file
  - http://pages.cpsc.ucalgary.ca/~tamj/resources/Verifying_D2L_Submissions.pdf

Backing Up And Submitting Your Work

- Bottom line: **it is up to you** to make sure things are done correctly and on time.
- If you have questions beforehand then do ask (make sure you ask your questions early enough so you can receive an answer before the due time).
- But don't wait until after the due date (it's too late).
  - If your work isn't in D2L before the due date then you will be awarded no credit.

Mini Assignments

- There will be four mini assignments worth 0.5/100 of the term grade for a total of 2/100 of term grade.

- The focus is learning how to apply the technical concepts (e.g., branches, functions, loops etc.) by writing a small and relatively simple program.

- Marking will focus on 'functionality': getting the program to work

- Although you shouldn't ignore other things such as style and documentation these things won't be graded for the mini-assignments

Full Assignments

- Similar to the mini assignments you will write a computer program
  - The programs will be larger and more challenging than the mini-assignments (require a 'tough' problem to be solved).

- Marking will not only be based on the functionality of your program but other criteria such as programming style and documentation (additional details will be provided during the semester as each assignment is released).

- Total weight of the full assignments *33/100 of term grade*
  - Assignment 1: *worth 4/100 (introduction to UNIX and writing programs)*
  - Assignment 2: *worth 6/100 (introduction to simple programming concepts)*
  - Assignment 3: *worth 6/100 (moderately challenging)*
  - Assignment 4: *worth 10/100 (quite challenging)*
  - Assignment 5: *worth 7/100 (quite challenging but in a different way than the previous assignment)*

Examinations

- There will be two examinations: midterm and final exam.
- Midterm exam (if a multiple section semester then it will be common to all lectures, out of class)
  - Proportion of the term grade: 25/100
  - **In class midterm Tuesday June 6**
- Final exam (again for multi-section: common to all lectures, out of class)
  - Proportion of the term grade: 40/100
  - Date/time/location determined by the Office the Registrar.
  - You can find information about your final exams online via the university PeopleSoft portal.
- Both exams will completed on paper (not in front of a computer).
- **You must pass the weighted average of the exam component to be awarded a grade of C- or higher in the class.**

James Tam

Examinations (2)

- Information about the examinations will be available on the main grid before the respective test:
  - Under the main index:
    - Main grid: Course topics, lecture notes, assignment descriptions, exam information

Course web page: Introduction to Computer Science for majors I James Tam

CPSC 231: Spring 2017

Index

- Lecture and important assignment information
- Tutorial information
- Course topics/notes, assignment/exam information

		[Acrobat notes] [PowerPoint notes] [Example programs]		
Lecture and imp	May 28 - June 3	**Looping/repetition** (continued)		**Mini A2 due Wednesday May 31**
		Functions/decomposition [Acrobat notes] [PowerPoint notes] [Example programs]	Functions: Chapter 5 (don't worry about the material in Section 5.9, 5.10 yet)	**Full A2 due Friday June 2**
Day/Time L0	June 4 - 10	**Functions/decomposition** (continued)		**Mini A3 due Monday June 5**
		Composites (Strings, lists and a little on tuples) [Acrobat notes] [PowerPoint notes] [Example programs]	Composites, Lists & Tuples Chapter 7	**In class midterm Tuesday June 6: [Information link provided here sometime before the exam]**
			Composites, Strings: Chapter 8	

Examination Content

- •Multiple choice questions:
  - Partial program traces e.g., what's the program output
  - Basic program structure e.g., find the errors, which function or operator is needed for a particular mathematical operation
  - More examples and details coming during the semester

- •Written questions:
  - Write a small/partial computer program.
  - Trace the execution of a computer program e.g., what is the 'output'.
  - Conceptual (lower weight for this type of question) e.g., definition of a technical term.
  - Likely there will be a smaller proportion of written questions on the midterm vs. the final.

- •I will be grading the exams.
  - (I'll do the best I can to get them done in a timely fashion but remember there's a fair number of you in the class).

James Tam

Examination Content (2)

- •More sample 'exam type' questions will be provided during the semester.
  - Sometimes 'on the fly' in lecture so **pay attention** to these and **take notes**.

James Tam

Grades For Each Component

- The official grading mechanism for this (and most) universities is a letter grade/grade point e.g. A/4.0, A-/3.7 etc.
- Term grades must be stated as a letter grade.
- Component grades (assignment, exam etc.) can either be a letter grade or a raw score (e.g. percentage)
- For this class
  - **each major component will be awarded a grade point (and not a percentage)…e.g. the 2.0 GPA and not 65% will be used to calculate your term grade.**
  - and this is the value used to determine the term grade.

Grading: Course Components

- Each course component will have a grading key
- Sample from a past assignment (different course)

5. Print the document (you won't actually be able to print anything in the 203 labs because there are no printers connected to the computers) but your program should be able to invoke the print command using VBA. (**+0.2 GPA**)

6. Close the document and automatically save changes (no choice given to the user). (**+0.2 GPA**)

7. Instead of applying Features 1 - 6 on just a single document, the macro will instead it will prompt the user for a location (e.g., "C:\temp") via a InputBox and apply Features 1 - 6 to every Word document in that location. When you write the program you can assume that the folder only contains Word documents. You must employ nesting in order to get credit for this feature, an outer loop successively opens each document in the specified location and inside the loop body Features 1 - 6 will be applied. (**+1.0 GPA**)

  - A student who completes only Features #5 – 7 will be awarded an assignment grade point of 0.2 + 0.2 + 1.0 = 1.4 (just over a D+ grade)

- Examinations will include a lookup table (raw percentage score to grade point). Sample from another course.

Min percentage	GPA	
85.00	3.8	
86.00	3.9	
87.00	4	A
94.00	4.1	
96.00	4.2	

Why Grade Points?

- It's the official university grading system
  - Alternatives are possible but require faculty level approval

- Approval of anything other than a grade point system requires predetermined cutoffs at the start of the term e.g., >= 90% equals 'A' etc.
  - Doesn't allow for consideration that individual components may be more challenging than others (lower cutoffs)

- Grade points are more lenient for grades on the lower-middle end of the scale
  - Grade points: Getting an "A"/4.0 on the assignment component worth 30% of the term grade yields a minimum term grade of 1.2 (4.0 * 0.3) which equates to a term grade of 'D' (possibly higher)
  - Percentages: Getting an "A" may roughly work out to 90% or higher (depending on the scale) which works out to a minimum term percent of 27% = 90% score * 30% weight…almost certainly an "F" for the term grade.

Mapping Raw Scores To Grade Points: Assignments

- For simplicity assignment marking keys will specify grade points only
  - Example (purely for illustration purposes):
    - Program features completed:
      - computes average sales = 0.3 grade points,
      - graphs results = 0.1 grade points etc.,
      - the user can re-run calculations with different data = 1.0 grade points.
  - Sum the grade points for each feature in order to yield the grade point awarded for the assignment e.g., 0.3 + 0.1 + 1.0 = 1.4 GPA earned (not 1.4/4.0 = 35%)

Mapping Raw Scores To Grade Points: Exams

- For examinations the mapping between a raw score and a grade point occurs one way (raw score mapped to grade point)
  - Example (purely for illustration purposes) 65 – 69% = C/2.0, 70 – 74% = C+/2.3
  - But grade points don't correlate back to percentages
    - e.g. I was awarded a 66% on midterm and then I see this is a 2.0 GPA (out of 4.0)
    - Does this mean that my percentage 'went' from a 66% to a 50%!!!???
    - No.
      - A C/2.0 does not mean that 50% was awarded as a course grade.
      - To put this in perspective a passing grade point in this university is a 1.0/D in a course. If a grade point mapped back to a percentage this would mean that anyone getting a 25% or higher would pass any course here.
  - The mapping of the midterm to grade point will be posted sometime after the midterm grades have been released.
  - The mapping of the final exam to grade point will be posted sometime after the final exam grades have been released.

James Tam

Estimating Your Overall Term Grade Point

- To determine your weighted term grade point simply *multiply each grade point* by the weight of each component.
  - **Percentages won't be used to determine the term grade point/letter**
- Sum the weighted grade points to determine the term grade.
- Simple and short example (not exactly the same as this term but it should be enough to give you an idea of how to do the specific calculations required this semester):
  - Assignments: weight = 30/100, example score = A
  - Midterm: weight = 30/100, example score = B+
  - Final: weight = 40/100, example score = C-
  - Weighted assignments: 0.3 * 4.0 = **1.2**
  - Weighted midterm: 0.3 * 3.3 = **0.99**
  - Weighted final: 0.4 * 1.7 = **0.68**
  - Total term grade point = **1.2** + **0.99** + **0.68** = **2.87**
  - (In this case the term letter is B-)

James Tam

Estimating Your Overall Term Grade Point (2)

- Use the spreadsheet on the course web page to estimate your term letter grade:
  - http://pages.cpsc.ucalgary.ca/~tamj/2017/203P/grade_calculator.xlsx
- The grade point to letter grade mapping employs the official university cutoffs:
  - http://www.ucalgary.ca/pubs/calendar/current/f-2.html
  - (I **may** employ a more lenient set of cutoffs at the end of term but the official cutoffs will provide you with a 'worse case' estimate of your grade).
  - Also an "A+" doesn't have a specified letter grade that is different from an "A" for term letter grades.
    - I'll likely extrapolate the pattern e.g. C+ = 2.3, B+ = 3.3 so an A+ will be higher than a 4.0 and a bit lower than 4.3 (because not all course components allow for an A+ to be awarded)

1 Note: to keep things simple the formula in the spreadsheet does not check if the exam component was passed or not (you can do the check manually or add it in yourself)

Common Computer Skills Assumed

- You know what a computer is!
- You've used a computer in some form (e.g., turn on, turn off, open a file, played a game, gone online etc.)
- You have experience *using common applications* (specifically email, web browsers, text editing using a word processor).

What This Course (CPSC 231) Is About

• Writing/creating computer programs.

• But it is *not assumed* that you have prior knowledge of Computer Science (or even experience writing programs)

• It can be a lot of work.

Late night 'coding'

Satisfaction coming from solving that tough algorithm!

- The course can be completed by students with a normal course load (many already have gotten through it!)
- But be cautious if you already have many other commitments

Actual Practice: Common Interview Questions

• Besides looking at degrees granted and grades received, some tech companies (e.g., Google) may ask you questions that appear non-technical:
  - You're asked to solve puzzles during the interview.
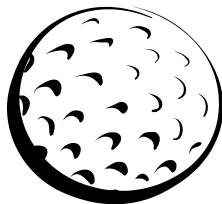
• There is a relationship between skill at solving puzzles ("problem solving") and success in a (technically oriented) industry.
  - You will develop these skills writing programs for this class.

Course Goals

- Know the basic structure of a computer program (rules for laying out a program and how the basic constructs such as repetition and branching work)
  - If you don't know and understand these concepts then your program won't work at all and you can't proceed to the next goals.
- Develop basic problem solving and analysis skills.
  - As mentioned this is a skill that you will need to develop for "the real world"
- Learn good design principles.
  - For example you may know how to get a program to run across the Internet but you may not know how to write a fun game that people will want to play on Facebook™.
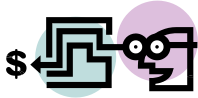 - "*This *%$#! App really sucks!*"

How To Succeed

- Successful people

How To Succeed In This Course

1. Practice things yourself (not by getting the answer from someone/someplace else).

Providing solutions to assignments may be popular among students but useless for learning

What's needed is for me to teach you the skills to solve any reasonable size problem

- How Computer Science works: You get better by doing things for yourself (this is a 'hands-on' field of study and work).

Similar to getting fit: you can't just watch

You have to do it yourself

How To Succeed In This Course (2)

- Write lots programs.
  - At the *very least* attempt every assignment.
  - Try to do some additional practice work (some examples will be given in class, some practice assignments will be available on the course web page).
  - Write lots of little 'test' programs to help you understand and apply the concepts being taught.

- Trace lots of code (computer programs)
  - Involves reading through programs that other people have written, and executing it 'by hand' in order to understand how and why it works
  - This is an essential skill.
  - Relying on just running the program and observing the results won't always work (errors?)
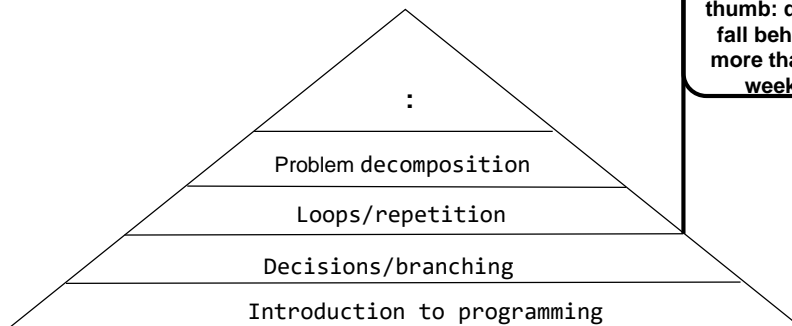
How To Succeed In This Course (3)

2. Make sure that you keep up with the material
    - Many of the concepts taught later depend upon your knowledge of earlier concepts.
    - Don't let yourself fall behind!
    - *At least* attempt all assignments!

Rule of thumb: don't fall behind more than 1 week

```
                 :
        Problem decomposition
          Loops/repetition
         Decisions/branching
      Introduction to programming
```

How To Succeed In This Course (4)

- If you find concepts unclear trying to research the answer on your own can be beneficial (because this is a 'hands on' field).
  - Read alternate explanations of the concepts covered in class in the text book (or other textbooks: remember that electronic books accessible through the library-Safari are 'free').
  - Looking at online resources:
    - Remember academic resources online just like other online information may not always be a good source.
    - Start with more reputable sources
      - http://proquest.safaribooksonline.com.ezproxy.lib.ucalgary.ca/
      - www.python.org
- Addendum to the previous point #2 and a point raised earlier "ask questions".
  - If you are still unclear on concepts then make sure that you ask for help.
  - Don't wait too long (more than a few days) to do this because latter concepts may strongly depend on your understanding of earlier concepts..

How To Succeed In This Course (5)

3. Look at the material before coming to lecture so you have a rough idea of what I will be talking about that day:
   a) Read the slides
   b) Look through the textbook(s)

When we get to more complicated programs that appear to 'jump around' in how they execute ("section: problem decomposition/functions") just having an idea of the scope and components of the program beforehand can be useful when I cover it in class.

How To Succeed In This Course (6)

4. Start working on things as early as possible:
   - Don't cram the material just before the exam, instead you should be studying the concepts as you learn them throughout the term.
   - It's important to work through and understand concepts *before* you start (full) assignments.
   - If you try to learn a new concept *and* work out a solution for the assignment at the same time then you may become overwhelmed.
   - Don't start assignments the night (or day!) that they are due, they may take more time than you first thought (start as soon as possible).
   - Some assignments may require the application of multiple concepts, not all the concepts have to be completely covered before you start working on an assignment.
     • Start working based on what's currently been covered (this will teach you how to decompose a program and work on it a part at a time).

How To Succeed In This Course: A Summary

1. Practice things yourself
2. Make sure that you keep up with the material
3. Look at the material before coming to lecture
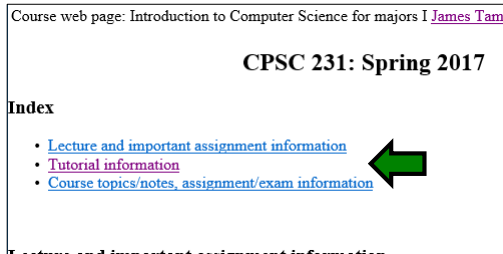4. Start working on things early

Computer Science: Labs And Tutorials (Reminder)

- Help tutorials ("Continuous Tutorial/CT"):
  - Attendance is not required (no official registration)
  - Q & A session: it will be used as an additional place where you can get help.
  - Located near the technical "Help Desk"
  - The CT schedule will be posted early in the semester.

- Teaching tutorials:
  - They will be conducted by the Teaching Assistants (TA).
  - A mandatory component of the course (registration in a specific section is required).
  - Review of concepts covered in lecture (especially the more challenging ones).
  - Discussion of assignment requirements.
  - Assignment and exam feedback/return after grading is complete.

Computer Science: Labs And Tutorials (Reminder: 2)

- •(Teaching tutorial information continued):
  - Practice exercises.
  - 'Open tutorials' will sometimes be held (extra CT/help time where TA's will be available to help students).

- •More information about tutorials and labs is available on the course web site:
  - http://pages.cpsc.ucalgary.ca/~tamj/2017/231P/index.html#Tutorial_information

Course web page: Introduction to Computer Science for majors I James Tam

CPSC 231: Spring 2017

Index

- Lecture and important assignment information
- Tutorial information
- Course topics/notes, assignment/exam information

Lecture and important assignment information

James Tam

Feedback

What is he talking about???

Wow I am the greatest speaker in the world!

Let me know how things are going in the course:
- Am I covering the material too slowly or too quickly.
- Can you read the slides and my hand writing.
- Can you hear me in the class.
- Etc.

James Tam