CSc 120
Introduction to Computer Programming II

00: Basic info
Welcome to CSc 120

• Second programming class
  – prerequisite: CSc 110 (or some programming experience)
  – looks at
    o more complex programs and programming problems
    o how data are represented and manipulated
    o how to think about and understand program behavior
  – start building the toolbox of a computer scientist

• Assumes you have at least a little programming experience
  – can write small programs; execute, test, and debug them
  – if not: take CSc 110
Instructional staff

• Instructor: Janalee O'Bagy, Ph.D.
  – Office: Gould-Simpson room 854
  – Email: jobagy@email.arizona.edu
  – Office hours:  Mon,Wed 1:00 – 2:30pm
    Tue 9:00 – 10:00am
    or if my door is open
    or by appointment (send email; put CS 120 in Subject:)

Check the website for updates to office hours
Instructional staff

• Section Leaders (SLs)
  – Introductions!

• SL contact info
  – Posted on the website

• SL GS 228 lab hours
  – Posted on Piazza
Meet your neighbors (2 minutes)

• Find out where they are from

• Together, decide on answers to these questions
  – What year was Python created?
  – How many websites are there today and how many were there 20 years ago?
  – The Python we use is written in C. How many lines of C code do you think it takes to implement Python?
Meet your neighbor (2 minutes)

• Find out where they are from
• Together, decide on answers to these questions
  – What year was Python created?
    o first posted in February 1991
  – How many websites are there today and how many were there 20 years ago?
  – The Python we use is written in C. How many lines of C code do you think it takes to implement Python?
Meet your neighbor (2 minutes)

• Find out where they are from
• Together, decide on answers to these questions
  – What year was Python created?
    o first posted in February 1991
  – How many websites are there today and how many were there 20 years ago?
    o today: 1.5 billion! (only 200 million are active)
    o 1998: 2.4 million (Google was launched in 98)
  – The Python we use is written in C. How many lines of C code do you think it takes to implement Python?
Meet your neighbor (2 minutes)

• Find out where they are from
• Together, decide on answers to these questions
  – What year was Python created?
    o first posted in February 1991
  – How many websites are there today and how many were there 20 years ago?
    o today: 1.5 billion! (only 200 million are active)
    o 1998: 2.4 million (Google was launched in 98)
  – The Python we use is written in C. How many lines of C code do you think it takes to implement Python?
    o Approx. 550,000 lines, including comments
Basic info about this class

• Programming language: **Python**
  – we will use Python 3
  – first lectures: review basics

• Development environment: **Idle**
  – comes with Python

• If you don’t know Python:
  – need to pick up the basics quickly
    o **make use of office hours!**
    o GS 228 lab and CS Tutoring Center in GS 942!
Are you in the right class?

• You have the pre-requisite
  – 110
  or
  – AP classes
  – ECE 175
  – ISTA 130
  – Community college transfer credits

• But...
  – At this point in your development, learning a new language is not easy
Are you in the right class?

• We want you to succeed!
• Use the first assignment as an assessment
  – there will be time to switch to 110
  – getting a strong foundation is the key to success in the CS program
• Flip side
  – 110 students, first week will be review for you
Course communication

• Piazza
  – piazza.com/arizona/spring2019/csc120
  – Sign up if you haven't already!
  – Questions are posted and answered here
  – Class communication takes place here

• Class Website
  – Important links: assignments, email contacts, syllabus, etc.
  – http://www2.cs.arizona.edu/classes/cs120/spring19/
  – See Piazza for updated office hours
Textbook

• Hardcopy
  *Problem Solving with Algorithms and Data Structures using Python (2nd ed.)*, by Bradley Miller and David Ranum.

• Online
  – [https://runestone.academy/runestone/static/pythonds/index.html](https://runestone.academy/runestone/static/pythonds/index.html)

• Additional resources
  – given as needed
  – plenty of additional on-line resources available
  – [https://docs.python.org/3](https://docs.python.org/3)
Collaborative learning

• Why are we in this classroom?
  – research shows that students learn more effectively when they are actively engaged

• Groups
  – in-class activities will give you the chance to work with your table group
  – no grade component associated with the group
  – group seating will be assigned later
EXERCISE

Work with your group to answer questions 1-7 in the Class Policies exercise handout.
EXERCISE

Work with your group to answer questions 1-7 in the Class Policies exercise handout.

1. b) False
2. b) False
3. c) 5%
4. a) once every week
5. b) False
6. f) All of the above if I have not used all of my late days
7. c) 7 days
In-class activities and Section Activities

• In-class activities (ICAs):
  – Activities (problems sets or quizzes) that are graded
  – Work in groups (except when noted)
  – Attendance not required, but ICAs count towards grade
    o ICAs are 5% of your grade
    o ICAs cannot be made up

Ans: 1. b) False, 2. b) False
In-class activities and Section Activities

• In-class activities (ICAs):
  – Activities (problems sets or quizzes) that are graded
  – Work in groups (except when noted)
  – Attendance not required, but ICAs count towards grade
    o ICAs cannot be made up
    o ICAs are 5%

• Sections:
  – Attendance is required
  – Section activities are graded for participation, not correctness
  – Sections are 5%

Ans: 3. c) 5%
Assignments

• Normal schedule
  – given out on Wed
    o several small problems:
      • auto-grader
      • due Saturday at 7:00pm
    o one or two larger problems:
      • due following Thursday at 7:00pm
  – graded feedback back to you by following Monday

• ~ 12 assignments over the semester

• one assignment per week (no assignments in exam weeks)
  Ans: 4. a) once every week

• no assignments are dropped
  – do them all!
Assignments

• Due at time specified
  – 3 late days (long problems only)
    Ans: 5. b) False
  – may submit up to 24-hours late
    Ans: 6. f) All of the above if I have not used all of my late days
  – in almost all cases: no extensions

➢ plan ahead
  o plan around assignments for other classes, other commitments

➢ start early
  o procrastination is the surest way to sabotage your performance in this class
Assignments

• Regrade requests
  – must be requested within 7 days of receiving feedback
  Ans: 7. c) 7 days
Assignments

• Grading:
  – coding style
    o code structure, comments, etc.
  – functionality
    o tested using a computer program
    o you need to follow directions exactly
      • file names
      • function names
      • input/output format
      • ... anything else specified...

Taking liberties with assignment specs is not the right way to show your creativity!
Exams

• Two midterms
  – approx. 5 weeks apart
    o see syllabus for dates
    o count for 30% of final grade (2 x 15%)

• Final exam:
  – See syllabus for dates
  – counts for 15% of final grade
Midterms

• In-class students
  – start at the beginning of lecture period
• 50 mins each
• No make-up exams except for unforeseeable emergencies
Grading policy

Components of your final grade:

<table>
<thead>
<tr>
<th>Component</th>
<th>Weight</th>
</tr>
</thead>
<tbody>
<tr>
<td>programming assignments</td>
<td>45%</td>
</tr>
<tr>
<td>in-class activities</td>
<td>5%</td>
</tr>
<tr>
<td>weekly section participation</td>
<td>5%</td>
</tr>
<tr>
<td>midterms</td>
<td>30%</td>
</tr>
<tr>
<td>final exam</td>
<td>15%</td>
</tr>
</tbody>
</table>

Regrade request deadlines:
- programs: within 1 week of getting grade back
- midterms: within 1 week of getting grade back
Grading policy

Grade boundaries:

<table>
<thead>
<tr>
<th>Percentage Range</th>
<th>Grade</th>
</tr>
</thead>
<tbody>
<tr>
<td>90% and above:</td>
<td>A</td>
</tr>
<tr>
<td>80% and above, but below 90%:</td>
<td>B</td>
</tr>
<tr>
<td>70% and above, but below 80%:</td>
<td>C</td>
</tr>
<tr>
<td>60% and above, but below 70%:</td>
<td>D</td>
</tr>
<tr>
<td>Below 65%:</td>
<td>E</td>
</tr>
</tbody>
</table>

(I may lower the cutoffs but will not raise them.)
Behavior and conduct

• treat each other with respect and courtesy
• don't be disruptive

• these behaviors will not be tolerated in class:
  – phone conversations, texting
  – reading newspapers or magazines
  – games, facebook, other social media
  – extended conversations (not class related)

please leave the room if you have to do any of these activities; come back when done.
Academic integrity

• Any work submitted for credit must be your own work
  – OK:
    o general discussions of how to approach a problem
  – NOT OK:
    o discussing the specifics of the code for an assignment
    o partnering with someone else on an assignment
    o soliciting help on online forums (e.g., stackoverflow)
Academic integrity

• Helping someone else cheat is just as bad as cheating yourself:
  – don't show your code to anyone else
  – don't share details of code ahead of a submission deadline
  – don't post your assignment code publicly
    o this includes Piazza
    o see me or the SLs to discuss your code

• See syllabus for detailed list of do's and don’t's
How to succeed in this class

• Understand the material
  – if you don’t: *ask questions!*
  – office hours

• Attend lecture and section
  – *participate!*

• Do the programming assignments
  – start early (only 3 late days)
  – follow directions *exactly*
  – test your code thoroughly
  – *don’t forget to submit your code!*
Expectations for next time

• Video
  – D2L -> Tools -> Panopto
  – Getting started with Python (23 min.)
  – Watch before Friday's lecture
  – Covers posted Python review slides up to 40
Expectations for next time

• Required survey
  – Survey link posted on Piazza
  – Due Saturday, at 7:00pm
  – It will take 5 minutes
Answers to the remaining Class Policies questions
8. e) pgm_file.py
9. b) False
10. f) None of the above
11. a) True
12. d) myfunc(num, string)
13. a) True
14. d) Never