

CompSci 101

Stable Sorting, Lambda

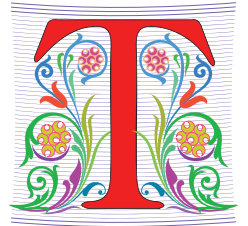
```
f = lambda x : x[1]
sorted(lst, key=f)
```

Susan Rodger
April 4, 2023

4/4/23

Compsci 101, Spring 2023 1

T is for ...



- **Type**
 - From int to float to string to list to ...
- **Text**
 - From .txt to editors to ...
- **Turing Award – Highest Honor in CS**
 - Nobel, Fields, Turing
 - Turing Duke Alums:
 - Ed Clarke (MS)
 - John Cocke (BS, PhD)
 - Fred Brooks (BS)

4/4/23

Compsci 101, Spring 2023

2

Shaundra Daily



- **Professor of the Practice, Duke University**
- **B.S. Florida State, Electrical Eng**
- **PhD Media Arts/Sciences – MIT**
- **Combines Dance with Robotics**
- **Focuses on technologies, programs and curricula to support Diversity, Equity and Inclusion in STEM Fields**



4/4/23

Compsci 101, Spring 2023 3

Announcements

- **Assignment 5 due Thursday!**
 - Sakai quiz due tonight! (no grace day)
- **Assignment 6 out Thursday, due April 20**
- **APT-6 out today, Due 4/13**
- **Still to come (APT-7 and Assign 7 (short))**
- **Lab 9 Friday**
 - There is a prelab
- **Coming up...**
 - Exam 3 – Tues, April 11

4/4/23

Compsci 101, Spring 2023 4

Exam 3– Tues, April 11 – in one week!

- **Exam is in class on paper – 10:15am**
 - Need pen or pencil
- **See materials under 4/11 date**
 - Exam 3 Reference sheet - part of exam
- **Covers**
 - topics
 - APTs through APT6
 - Labs through Lab 9
 - Assignments through Assignment 5

Tuesday
4/11
No Reading No QZ
*** EXAM 3 ***
Recommended Old Tests
Exam 3 Reference Sheet
All Old tests

11/17/22

Compsci 101, Fall 2022

5

Exam 3

- **Exam 3 is your own work!**
- **No looking at other people's exam**
- **You cannot use any notes, books, computing devices, calculators, or any extra paper**
- **Bring only a pen or pencil**
- **The exam has extra white space and has the Exam 3 reference sheet as part of the exam.**
- **Do not discuss any problems on the exam with others until it is handed back**

11/17/22

Compsci 101, Fall 2022

7

Exam 3 topics include ...

- **List, tuples, list comprehensions**
- **Loops – for loop, while loop, indexing with a loop**
- **Reading from a file**
 - Converting data into a list of things
- **Parallel lists**
- **Sets – solving problems**
- **Dictionaries – solving problems**
- **Sorting – lists, tuples**
- **No turtles, no images - but note we are practicing other concepts with images**

11/17/22

Compsci 101, Fall 2022

6

Exam 3 – How to Study

- **Practice writing code on paper!**
- **Rewrite an APT**
- **Try to write code from lecture from scratch**
- **Try to write code from lab from scratch**
- **Practice from old exams**
- **Put up old Sakai quizzes, but better to practice writing code**
- **Look at Exam 3 reference sheet when writing code!**

11/17/22

Compsci 101, Fall 2022

8

PFTD

- **Sorting in Python and sorting in general**
 - How to use `.sort` and `sorted`, differences
 - Key function – change how sorting works
 - Lambda – create anonymous functions
- **Stable sorting**
 - How to leverage when solving problems
 - Why Timsort is the sort-of-choice (! quicksort)

API to change sorting

- **In SongReader.py we changed order of tuples to change sorting order**
 - Then we sliced the end to get "top" songs
- **Can supply a function to compare elements**
 - Function return value used to sort, `key=function`
 - Change order: `reverse=True`

Python Sorting API

- **We'll use both `sorted()` and `.sort()` API**
 - How to call, what options are
 - How to sort on several criteria
- **One creates a new list, one modifies existing list**
 - `sorted(..)` creates list from .. Iterable
 - `x.sort()` modifies the list x, no return value!

Sorting Examples (with optional parameters)

- **Use `key=function` argument and `reverse=True`**
 - What if we want to write our own function?

```
a = ['red', 'orange', 'green', 'blue', 'indigo', 'violet']  
print(sorted(a))
```

```
print(sorted(a, key=len))
```

```
print(sorted(a, key=len, reverse=True))
```

Sorting Examples (with optional parameters)

- Use `key=function` argument and `reverse=True`
 - What if we want to write our own function?

```
a = ['red', 'orange', 'green', 'blue', 'indigo', 'violet']
print(sorted(a))
['blue', 'green', 'indigo', 'orange', 'red', 'violet']
print(sorted(a, key=len))
['red', 'blue', 'green', 'orange', 'indigo', 'violet']
print(sorted(a, key=len, reverse=True))
['orange', 'indigo', 'violet', 'green', 'blue', 'red']
```

4/4/23

Compsci 101, Spring 2023 13

Sorting Examples

```
a = [4, 1, 7, 3]
b = sorted(a)
a.sort()
a = ['Q', 'W', 'B', 'F']
b = sorted(a)
c = sorted(a, reverse = True)
a = ['hello', 'blue', 'car']
b = sorted(a, key=len)
a: [4, 1, 7, 3]
b: [1, 3, 4, 7]
a: [1, 3, 4, 7]
a: ['Q', 'W', 'B', 'F']
b: ['B', 'F', 'Q', 'W']
c: ['W', 'Q', 'F', 'B']
a: ['hello', 'blue', 'car']
b: ['car', 'blue', 'hello']
```

4/4/23

Compsci 101, Spring 2023 15

Sorting Examples

```
a = [4, 1, 7, 3]
b = sorted(a)
a.sort()
a = ['Q', 'W', 'B', 'F']
b = sorted(a)
c = sorted(a, reverse = True)
a = ['hello', 'blue', 'car']
b = sorted(b, key=len)
```

4/4/23

Compsci 101, Spring 2023 14

More Sorting Examples

```
a = [ [2, 2, 34], [2, 6, 7, -1], [1, 2, 3] ]
b = sorted(a)
c = sorted(a, key = len)
d = sorted(a, key=max)
e = sorted(a, key=min)
```

4/4/23

Compsci 101, Spring 2023 16

More Sorting Examples

Sort lists
on first
element

Sort lists
on length
of lists

Sort lists
on max
element

Sort lists
on min
element

```
a = [ [2, 2, 34], [2, 6, 7, -1], [1, 2, 3] ]
b = sorted(a)
b: [ [1, 2, 3], [2, 2, 34], [2, 6, 7, -1] ]
c = sorted(a, key = len)
c: [ [2, 2, 34], [1, 2, 3], [2, 6, 7, -1] ]
d = sorted(a, key=max)
d: [ [1, 2, 3], [2, 6, 7, -1], [2, 2, 34] ]
e = sorted(a, key=min)
e: [ [2, 6, 7, -1], [1, 2, 3], [2, 2, 34] ]
```

4/4/23

Compsci 101, Spring 2023 17

WOTO – 1st question

Showing the list and the list sorted

```
In[14]: a = ['red', 'orange', 'yellow', 'green', 'blue', 'indigo', 'violet']
In[15]: sorted(a)
Out[15]: ['blue', 'green', 'indigo', 'orange', 'red', 'violet', 'yellow']
```

What's the list returned by sorted(a, reverse=True)? *

- ['yellow','violet', 'red', 'orange', 'indigo', 'green', 'blue']
- ['violet', 'indigo', 'blue', 'green', 'yellow', 'orange', 'red']

4/4/23

Compsci 101, Spring 2023 19

WOTO-1 Basic Sorting

<http://bit.ly/101s23-0404-1>

4/4/23

Compsci 101, Spring 2023 18

WOTO – 1st question

Showing the list and the list sorted

```
In[14]: a = ['red', 'orange', 'yellow', 'green', 'blue', 'indigo', 'violet']
In[15]: sorted(a)
Out[15]: ['blue', 'green', 'indigo', 'orange', 'red', 'violet', 'yellow']
```

What's the list returned by sorted(a, reverse=True)? *

- ['yellow','violet', 'red', 'orange', 'indigo', 'green', 'blue'] ←
- ['violet', 'indigo', 'blue', 'green', 'yellow', 'orange', 'red']

4/4/23

Compsci 101, Spring 2023 20

WOTO – 2cd question

Showing the list and the list sorted

```
In[14]: a = ['red', 'orange', 'yellow', 'green', 'blue', 'indigo', 'violet']
In[15]: sorted(a)
Out[15]: ['blue', 'green', 'indigo', 'orange', 'red', 'violet', 'yellow']
```

What's the list returned by sorted(a, key=len)? *

- ['red', 'blue', 'green', 'orange', 'yellow', 'indigo', 'violet']
- ['red', 'blue', 'orange', 'green', 'yellow', 'indigo', 'violet']

WOTO – 2cd question

Showing the list and the list sorted

```
In[14]: a = ['red', 'orange', 'yellow', 'green', 'blue', 'indigo', 'violet']
In[15]: sorted(a)
Out[15]: ['blue', 'green', 'indigo', 'orange', 'red', 'violet', 'yellow']
```

What's the list returned by sorted(a, key=len)? *

- ['red', 'blue', 'green', 'orange', 'yellow', 'indigo', 'violet'] ←
- ['red', 'blue', 'orange', 'green', 'yellow', 'indigo', 'violet']

WOTO – 3rd question

Showing the list and the list sorted

```
In[14]: a = ['red', 'orange', 'yellow', 'green', 'blue', 'indigo', 'violet']
In[15]: sorted(a)
Out[15]: ['blue', 'green', 'indigo', 'orange', 'red', 'violet', 'yellow']
```

The function max applied to a string returns the alphabetically greatest character in the string, so max('indigo') == 'o' and max('yellow') == 'y'. What's the list returned by sorted(a, key=max)? *

- ['indigo', 'orange', 'green', 'red', 'blue', 'violet', 'yellow']
- ['indigo', 'red', 'orange', 'green', 'blue', 'violet', 'yellow']

WOTO – 3rd question

Showing the list and the list sorted

```
In[14]: a = ['red', 'orange', 'yellow', 'green', 'blue', 'indigo', 'violet']
In[15]: sorted(a)
Out[15]: ['blue', 'green', 'indigo', 'orange', 'red', 'violet', 'yellow']
```

The function max applied to a string returns the alphabetically greatest character in the string, so max('indigo') == 'o' and max('yellow') == 'y'. What's the list returned by sorted(a, key=max)? *

- ['indigo', 'orange', 'green', 'red', 'blue', 'violet', 'yellow']
- ['indigo', 'red', 'orange', 'green', 'blue', 'violet', 'yellow'] ←

The power of lambda

- We want to create a function "on-the-fly"
 - aka anonymous function
 - aka "throw-away" function

```
In[7]: a
Out[7]: ['red', 'orange', 'green', 'blue', 'indigo', 'violet']
In[8]: sorted(a, key=lambda x : x.count("e"))
Out[8]: ['indigo', 'red', 'orange', 'blue', 'violet', 'green']
```

- Why 'indigo' first and 'green' last?
 - What about order of ties? Later today! Stable

Why is lambda used?

- It doesn't matter at all could use zeta? iota? ...
 - https://en.wikipedia.org/wiki/Alonzo_Church
- Lisp and Scheme have lambda expressions
- Guido van Rossum, learned to live with lambda



Anonymous Functions

- Useful when want "throw-away" function
 - Our case mainly sort
- Syntax: **lambda** PARAMETERS: EXPRESSION
 - PARAMETERS – 0 or more comma separated
 - EXPRESSION – evaluates to something

What is a lambda expression?

- It's a function object, treat like expression/variable
 - Like list comprehensions, access variables

```
>>> inc = lambda x : x + 1
>>> p = [1, 3, 5, 7]
>>> [inc(num) for num in p]
[2, 4, 6, 8]
```

Syntactic sugar (makes the medicine go down)

- Syntactic sugar for a normal function definition

```
def f(x):  
    return x[1]  
sorted(lst, key=f)
```

```
>>> d.items()  
dict_items([('a', [1, 2, 3]), ('b', [4, 7]), ('c', [1, 1, 5, 8])])  
>>> sorted(d.items(), key=lambda x : len(x[1]))  
  
>>> sorted(d.items(), key=lambda sparky : len(sparky[1]))
```

4/4/23

Compsci 101, Spring 2023 29

Syntactic sugar (makes the medicine go down)

- Syntactic sugar for a normal function definition

```
def f(x):  
    return x[1]  
sorted(lst, key=f)
```

```
f = lambda x : x[1]  
sorted(lst, key=f)
```

```
sorted(lst, key=lambda x : x[1])
```

```
>>> d.items()  
dict_items([('a', [1, 2, 3]), ('b', [4, 7]), ('c', [1, 1, 5, 8])])  
>>> sorted(d.items(), key=lambda x : len(x[1]))  
  
>>> sorted(d.items(), key=lambda sparky : len(sparky[1]))
```

Parameter
name does
not matter

4/4/23

Compsci 101, Spring 2023 30

Syntactic sugar (makes the medicine go down)

- Syntactic sugar for a normal function definition

```
def f(x):  
    return x[1]  
sorted(lst, key=f)
```

```
f = lambda x : x[1]  
sorted(lst, key=f)
```

```
sorted(lst, key=lambda x : x[1])
```

```
>>> d.items()  
dict_items([('a', [1, 2, 3]), ('b', [4, 7]), ('c', [1, 1, 5, 8])])  
>>> sorted(d.items(), key=lambda x : len(x[1]))  
[('b', [4, 7]), ('a', [1, 2, 3]), ('c', [1, 1, 5, 8])]  
>>> sorted(d.items(), key=lambda sparky : len(sparky[1]))  
[('b', [4, 7]), ('a', [1, 2, 3]), ('c', [1, 1, 5, 8])]
```

Parameter
name does
not matter

4/4/23

Compsci 101, Spring 2023 31

Syntax and Semantics of Lambda

- Major use: single variable function as key

```
fruits = ['banana', 'apple', 'lemon', 'kiwi', 'pineapple']  
b = sorted(fruits)
```

```
c = min(fruits)
```

```
d = max(fruits)
```

4/4/23

Compsci 101, Spring 2023 32

Syntax and Semantics of Lambda

- Major use: single variable function as key

```
fruits = ['banana', 'apple', 'lemon', 'kiwi', 'pineapple']
b = sorted(fruits)
  b: ['apple', 'banana', 'kiwi', 'lemon', 'pineapple']
c = min(fruits)
  c: 'apple'
d = max(fruits)
  d: 'pineapple'
```

4/4/23

Compsci 101, Spring 2023 33

Syntax and Semantics of Lambda (2)

```
fruits = ['banana', 'apple', 'lemon', 'kiwi', 'pineapple']

e = min(fruits, key=lambda f: len(f) )

g = max(fruits, key=lambda z: z.count('e') )

h = sorted(fruits, key=lambda z: z.count('e') )
```

4/4/23

Compsci 101, Spring 2023 34

Syntax and Semantics of Lambda (2)

```
fruits = ['banana', 'apple', 'lemon', 'kiwi', 'pineapple']

e = min(fruits, key=lambda f: len(f) )
  e: 'kiwi'

g = max(fruits, key=lambda z: z.count('e') )
  g: 'pineapple'

h = sorted(fruits, key=lambda z: z.count('e') )
  h: ['banana', 'kiwi', 'apple', 'lemon', 'pineapple']
```

4/4/23

Compsci 101, Spring 2023 35

Review: CSV and Sort for top artists

- Using two-sorts to get top artists

```
31 print('\nTop 5 artists:')
32 sortbycount = sorted([(a[1], a[0]) for a in counts.items()])
33 sortedArtists = [(a[1], a[0]) for a in sortbycount]
34 for artist in sortedArtists[-5:]:
35     print(artist)
```

- Reverse tuples to sort
- Reverse tuples to print

```
Top 5 artists:
('John, Elton', 21)
('Who', 24)
('Rolling Stones', 36)
('Led Zeppelin', 38)
('Beatles', 51)
```

4/4/23

Compsci 101, Spring 2023 36

Top 5 Artists

- Instead of intermediary list, use `lambda`
- Instead of `[-5:]`, use `reverse=True`

```
31 print('\nTop 5 artists:')
32 sortbycount = sorted([(a[1], a[0]) for a in counts.items()])
33 sortedArtists = [(a[1], a[0]) for a in sortbycount]
34 for artist in sortedArtists[-5:]:
35     print(artist)
36
37 print("repeat it")
38 sortedArtists = sorted(counts.items(), key=lambda item: item[1], reverse=True)
39 for tup in sortedArtists[:5]:
40     print(tup)
```

```
repeat it
('Beatles', 51)
('Led Zeppelin', 38)
('Rolling Stones', 36)
('Who', 24)
('Eagles', 21)
```

4/4/23

Compsci 101, Spring 2023

37

Top 5 Artists

- Instead of intermediary list, use `lambda`
- Instead of `[-5:]`, use `reverse=True`

```
31 print('\nTop 5 artists:')
32 sortbycount = sorted([(a[1], a[0]) for a in counts.items()])
33 sortedArtists = [(a[1], a[0]) for a in sortbycount]
34 for artist in sortedArtists[-5:]:
35     print(artist)
36
37 print("repeat it")
38 sortedArtists = sorted(counts.items(), key=lambda item: item[1], reverse=True)
39 for tup in sortedArtists[:5]:
40     print(tup)
```

Output slightly different. Why?

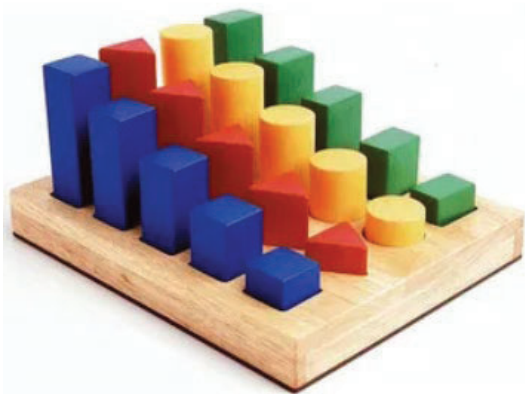
```
repeat it
('Beatles', 51)
('Led Zeppelin', 38)
('Rolling Stones', 36)
('Who', 24)
('Eagles', 21)
```

4/4/23

Compsci 101, Spring 2023

38

WOTO-2 Sorting
<http://bit.ly/101s23-0404-2>



4/4/23

Compsci 101, Spring 2023

39

That last question on the WOTO

- We haven't seen that yet!!!!!!!
- The tuple indicates how to sort and how to break ties.
- See code

4/4/23

Compsci 101, Spring 2023

40

What is happening?

```
f = open('twain.txt')
lines = [l.strip() for l in f]
s = sorted(lines, key=lambda line: (len(line), line))
```

Sorting this

Using this
criteria to sort

Using this
criteria to
break ties
alphabetically

4/4/23

Compsci 101, Spring 2023 41

How is the sorting happening?

```
>>> d
{'a': [1, 2, 3], 'b': [4, 7], 'c': [1, 1, 5, 8]}
>>> sorted(d.items())

>>> sorted(d.items(), key=lambda x: x[1])

>>> sorted(d.items(), key=lambda x: x[1][-1])
```

4/4/23

Compsci 101, Spring 2023 42

How is the sorting happening?

```
>>> d
{'a': [1, 2, 3], 'b': [4, 7], 'c': [1, 1, 5, 8]}
>>> sorted(d.items())
[('a', [1, 2, 3]), ('b', [4, 7]), ('c', [1, 1, 5, 8])]
>>> sorted(d.items(), key=lambda x: x[1])
[('a', [1, 2, 3]), ('b', [4, 7]), ('c', [1, 1, 5, 8])]
>>> sorted(d.items(), key=lambda x: x[1][-1])
[('c', [1, 1, 5, 8]), ('a', [1, 2, 3]), ('b', [4, 7])]
```

4/4/23

Compsci 101, Spring 2023 43

How is the sorting happening?

```
>>> d
{'a': [1, 2, 3], 'b': [4, 7], 'c': [1, 1, 5, 8]}
>>> sorted(d.items())
[('a', [1, 2, 3]), ('b', [4, 7]), ('c', [1, 1, 5, 8])]
>>> sorted(d.items(), key=lambda x: x[1])
[('a', [1, 2, 3]), ('b', [4, 7]), ('c', [1, 1, 5, 8])]
>>> sorted(d.items(), key=lambda x: x[1][-1])
[('c', [1, 1, 5, 8]), ('a', [1, 2, 3]), ('b', [4, 7])]
```

4/4/23

Compsci 101, Spring 2023 44

How is the sorting happening?

```
>>> d
{'a': [1, 2, 3], 'b': [4, 7], 'c': [1, 1, 5, 8]}
>>> sorted(d.items())
[('a', [1, 2, 3]), ('b', [4, 7]), ('c', [1, 1, 5, 8])]
>>> sorted(d.items(), key=lambda x: x[1])
[('c', [1, 1, 5, 8]), ('a', [1, 2, 3]), ('b', [4, 7])]
>>> sorted(d.items(), key=lambda x: x[1][-1])
[('a', [1, 2, 3]), ('b', [4, 7]), ('c', [1, 1, 5, 8])]
```

4/4/23

Compsci 101, Spring 2023 45

How to do some “fancy” sorting

- **lambda PARAMETER : EXPRESSION**
- **Given data: list of tuples: (first name, last name, age)**
[('Percival', 'Avram', 51), ('Melete', 'Sandip', 24), ...]
- **What does this do?**
- **sorted(data, key=lambda z : (z[0],z[1],z[2]))**

- **What is the lambda key to sort the following?**
 - Sort by last name, break ties with first name
 - Sort by last name, break ties with age
 - Alphabetical by last name, then first name, then reverse age order

4/4/23

Compsci 101, Spring 2023 46

How to do some “fancy” sorting

- **lambda PARAMETER : EXPRESSION**
- **Given data: list of tuples: (first name, last name, age)**
[('Percival', 'Avram', 51), ('Melete', 'Sandip', 24), ...]
- **What does this do?**
- **sorted(data, key=lambda z : (z[0],z[1],z[2]))**
 - **Sorts by first name, break ties with last name, break further ties with age**
- **What is the lambda key to sort the following?**
 - Sort by last name, break ties with first name
 - Sort by last name, break ties with age
 - Alphabetical by last name, then first name, then reverse age order

4/4/23

Compsci 101, Spring 2023 47

Creating Tuples with lambda

- **Sort by last name, break ties with first name**
- **Sort by last name, break ties with age**
- **Alphabetical by last name, then first name, then reverse age order**

4/4/23

Compsci 101, Spring 2023 48

Creating Tuples with lambda

- **Sort by last name, break ties with first name**
 - `key = lambda x: (x[1], x[0])`
- **Sort by last name, break ties with age**
 - `key = lambda x: (x[1], x[2])`
- **Alphabetical by last name, then first name, then reverse age order**
 - `key = lambda x: (x[1], x[0], -x[2])`
- **What if wanted something really different?**
 - Sort alphabetical by last name, break ties by reverse alphabetical using first name

Minus "-" means reverse, can only use with numbers

Can't use "minus" with strings

Leveraging the Algorithm

- **Can't sort by creating a tuple with lambda, use:**
 - Pattern: Multiple-pass *stable* sort – first sort with last tie breaker, then next to last tie breaker, etc. until at main criteria
- **Sort by index 0, break tie in reverse order with index 1**
[('b', 'z'), ('c', 'x'), ('b', 'x'), ('a', 'z')]
- **Stable sort respects original order of "equal" keys**

Leveraging the Algorithm

- **Can't sort by creating a tuple with lambda, use:**
 - Pattern: Multiple-pass *stable* sort – first sort with last tie breaker, then next to last tie breaker, etc. until at main criteria
- **Sort by index 0, break tie in reverse order with index 1**
[('b', 'z'), ('c', 'x'), ('b', 'x'), ('a', 'z')]
[('b', 'z'), ('a', 'z'), ('c', 'x'), ('b', 'x')]
- **Stable sort respects original order of "equal" keys**

Leveraging the Algorithm

- **Can't sort by creating a tuple with lambda, use:**
 - Pattern: Multiple-pass *stable* sort – first sort with last tie breaker, then next to last tie breaker, etc. until at main criteria
- **Sort by index 0, break tie in reverse order with index 1**
[('b', 'z'), ('c', 'x'), ('b', 'x'), ('a', 'z')]
[('b', 'z'), ('a', 'z'), ('c', 'x'), ('b', 'x')]
[('a', 'z'), ('b', 'z'), ('b', 'x'), ('c', 'x')]
- **Stable sort respects original order of "equal" keys**

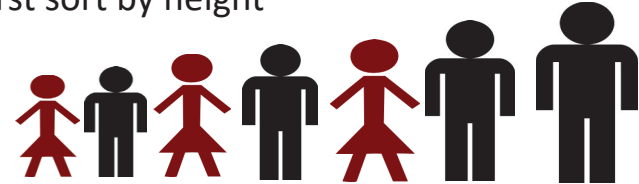
Stable sorting: respect "equal" items

- Women before men, each group height-sorted
 - First sort by height



Stable sorting: respect "equal" items

- Women before men, each group height-sorted
 - First sort by height



- Then sort by gender



4/4/23

Compsci 101, Spring 2023 53

4/4/23

Compsci 101, Spring 2023 54

Understanding Multiple-Pass Sorting

```
> data
[('f', 2, 0), ('e', 1, 4), ('a', 2, 0),
 ('c', 2, 5), ('b', 3, 0), ('d', 2, 4)]
> a0 = sorted(data, key = lambda x: x[0])
> a0

> a1 = sorted(a0, key = lambda x: x[2])
> a1

> a2 = sorted(a1, key = lambda x: x[1])
> a2
```

4/4/23

Compsci 101, Spring 2023 55

Understanding Multiple-Pass Sorting

```
> data
[('f', 2, 0), ('e', 1, 4), ('a', 2, 0),
 ('c', 2, 5), ('b', 3, 0), ('d', 2, 4)]
> a0 = sorted(data, key = lambda x: x[0])
> a0
[('a', 2, 0), ('b', 3, 0), ('c', 2, 5),
 ('d', 2, 4), ('e', 1, 4), ('f', 2, 0)]
> a1 = sorted(a0, key = lambda x: x[2])
> a1

> a2 = sorted(a1, key = lambda x: x[1])
> a2
```

4/4/23

Compsci 101, Spring 2023 56

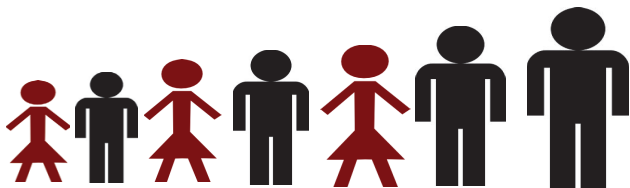
Understanding Multiple-Pass Sorting

```
> data
[('f', 2, 0), ('e', 1, 4), ('a', 2, 0),
 ('c', 2, 5), ('b', 3, 0), ('d', 2, 4)]
> a0 = sorted(data, key = lambda x: x[0])
> a0
[('a', 2, 0), ('b', 3, 0), ('c', 2, 5),
 ('d', 2, 4), ('e', 1, 4), ('f', 2, 0)]
> a1 = sorted(a0, key = lambda x: x[2])
> a1
[('a', 2, 0), ('b', 3, 0), ('f', 2, 0),
 ('d', 2, 4), ('e', 1, 4), ('c', 2, 5)]
> a2 = sorted(a1, key = lambda x: x[1])
> a2
```

4/4/23

Compsci 101, Spring 2023 57

WOTO-3 Multipass Sorting
<http://bit.ly/101s23-0404-3>



4/4/23

Compsci 101, Spring 2023 59

Understanding Multiple-Pass Sorting

```
> data
[('f', 2, 0), ('e', 1, 4), ('a', 2, 0),
 ('c', 2, 5), ('b', 3, 0), ('d', 2, 4)]
> a0 = sorted(data, key = lambda x: x[0])
> a0
[('a', 2, 0), ('b', 3, 0), ('c', 2, 5),
 ('d', 2, 4), ('e', 1, 4), ('f', 2, 0)]
> a1 = sorted(a0, key = lambda x: x[2])
> a1
[('a', 2, 0), ('b', 3, 0), ('f', 2, 0),
 ('d', 2, 4), ('e', 1, 4), ('c', 2, 5)]
> a2 = sorted(a1, key = lambda x: x[1])
> a2
[('e', 1, 4), ('a', 2, 0), ('f', 2, 0),
 ('d', 2, 4), ('c', 2, 5), ('b', 3, 0)]
```

4/4/23

Compsci 101, Spring 2023 58

WOTO-3 Q1: Unpack from Inside out

```
sorted(sorted(sorted(lst, key=sum), key=min), key=max)
```

```
lst = [ [4, 6, 7], [5, 2], [3, 9], [6, 2, 9] ]
```

```
x = sorted(lst, key=sum)
```

```
y = sorted(x, key = min)
```

```
z = sorted(y, key=max)
```

4/4/23

Compsci 101, Spring 2023 60

WOTO-3 Q1: Unpack from Inside out

```
sorted(sorted(sorted(lst, key=sum), key=min), key=max)
```

```
lst = [ [4, 6, 7], [5, 2], [3, 9], [6, 2, 9] ]
```

```
x = sorted(lst, key=sum)
```

```
x: [ [5, 2], [3, 9], [4, 6, 7], [6, 2, 9] ]
```

```
y = sorted(x, key = min)
```

```
y: [ [5, 2], [6, 2, 9], [3, 9], [4, 6, 7] ]
```

```
z = sorted(y, key=max)
```

```
z: [ [5, 2], [4, 6, 7], [6, 2, 9], [3, 9] ]
```

Answer

WOTO-3 Q2

- A dog show needs to order how they will give the awards. The data is a list of tuples. The tuples are (category, breed, score). What is the sort order in Python if using stable sorting (1) all dogs in the same category go together, (2) within a category dogs are ordered by breed, (3) within a breed dogs are ordered by score where the highest score goes first?
- If you do multiple-pass sorting use: score, breed, category
- If you do it in one line with lambda use: category, breed, score with -