Interpreting Exceptions

UW CSE 190p

Summer 2012

download examples from the calendar

There are two ways of constructing a software design: One way is to make it so simple that there are obviously no deficiencies, and the other way is to make it so complicated that there are no obvious deficiencies.

Hoare

```
def friends(graph, user):
    """Returns a set of the friends of the given user, in
the given graph."""
    return set(graph.neighbors(user))

def friends_of_friends(graph, user):
    """Returns a set of friends of friends of the given
user, in the given graph. The result does not include the
user nor their friends """
    fof = set()
    f = friends(graph, user)
    for fren in f:
        friends = friends(graph, user)
        fof = fof | friend
    g = (fof - f)
    g.remove(user)
    return g
```

```
myval=["Mercutio"]
print friends_of_friends(rj, myval)
```

```
Traceback (most recent call last):
File "nx_error.py", line 41, in <module>
  print friends_of_friends(rj, myval)
File "nx_error.py", line 30, in friends_of_friends
  f = friends(graph, user)
File "nx_error.py", line 25, in friends
  return set(graph.neighbors(user))#
File "/Library/Frameworks/…/graph.py", line 978, in neighbors
  return list(self.adj[n])
```

see nx_error.py

Traceback: a description of the *stack*.

```
Traceback (most recent call last):
File "nx_error.py", line 41, in <module>
  print friends_of_friends(rj, myval)
File "nx_error.py", line 30, in friends_of_friends
  f = friends(graph, user)
File "nx_error.py", line 25, in friends
  return set(graph.neighbors(user))#
File "/Library/Frameworks/…/graph.py", line 978, in neighbors
  return list(self.adj[n])
```

Each *stack frame* in the stack is described by a

- filename
- line number
- function name

Further, the line itself is printed for convenience

see nx_error.py

```
myval=["Mercutio"]
print friends_of_friends(rj, myval)
```

```
Traceback (most recent call last):
File "nx_error.py", line 41, in <module>
  print friends_of_friends(rj, myval)
File "nx_error.py", line 30, in friends_of_friends
  f = friends(graph, user)
File "nx_error.py", line 25, in friends
  return set(graph.neighbors(user))#
File "/Library/Frameworks/…/graph.py", line 978, in neighbors
  return list(self.adj[n])
```

How many stack frames are referenced?

Where did the error actually get noticed?

Where was the original cause of the problem?

see nx_error.py

```python
# assume rj was defined previously and correctly

def friends(graph, user):
    """Returns the set of friends of user in graph"""
    return set(graph.neighbors(user))

friends = friends(rj, "Mercutio")
print friends
friends = friends(rj, "Juliet")
print friends
```

What will be the output?

see name_conflict.py

```
def friends_of_friends(graph, user):
    """Returns a set of friends of friends of the given
user, in the given graph. The result does not include the
user nor their friends """
    fof = set()
    f = friends(graph, user)
    for fren in f:
        friends = friends(graph, user) # name conflict
        fof = fof | friend
    g = (fof - f)
    g.remove(user)
    return g
```

Same root cause problem,
very different message

see name_conflict2.py

```
def friends(graph, user):
    """Returns the set of friends of user in graph"""
    return set(graph.neighbors(user))

friends = friends(rj, "Mercutio") # name conflict
print friends

def friends_of_friends(graph, user):
    """Returns a set of friends of friends of the given
user, in the given graph. The result does not include the
user nor their friends """
    fof = set()
    f = friends(graph, user)
    for fren in f:
        friend = friends(graph, user)
        fof = fof | friend
    g = (fof - f)
    g.remove(user)
    return g

print friends_of_friends(rj, “Mecutio”)
```

see name_conflict3.py

```
# Two errors -- which is thrown first?

print x  # undefined variable

 print "x"  # bad indentation
```

Python performs a *syntax check* of your code before it executes anything.

see syntax_error.py

```
def friends_of_friends(graph, user):
    """Returns a set of friends of friends of the given user, in
the given graph. The result does not include the user nor their
friends """
    fof = set()
    f = friends(graph, user)
    for fren in f:
        friend = friends(graph, user)
        fof = fof | friend
        fof = fof.remove(user)
    g = (fof - f)
    return g
```

```
Traceback (most recent call last):
  File "none_error.py", line 21, in <module>
    friends_of_friends(g, "Mercutio")
  File "none_error.py", line 13, in friends_of_friends
    fof = fof | friend
TypeError: unsupported operand type(s) for |: 'NoneType' and 'set'
```

see none_error.py

```python
def friends_of_friends(graph, user):
    """Returns a set of friends of friends of the given user, in
the given graph. The result does not include the user nor their
friends """
    fof = set()
    f = friends(graph, user)
    for fren in f:
        friend = friends(graph, user)
        fof = fof | friend
    g = (fof - f) - user
    return g
```

```
Traceback (most recent call last):
  File ”social_network.py", line 20, in <module>
    friends_of_friends(g, 2)
  File ”social_network.py", line 14, in friends_of_friends
    g = (fof - f) - user
TypeError: unsupported operand type(s) for -: 'set' and 'int'
```

see type_error.py

```
def friends_of_friends(graph, user):
    """Returns a set of friends of friends of the given user, in
the given graph. The result does not include the user nor their
friends """
    fof = set()
    f = friends(graph, user)
    for fren in f:
        friend = friends(graph, user)
        fof = fof | friend
    f.add(set([user]))
    g = (fof - f)
    return g
```

```
Traceback (most recent call last):
  File "unhashable_type.py", line 21, in <module>
    friends_of_friends(g, "Mercutio")
  File "unhashable_type.py", line 14, in friends_of_friends
    f.add([user])
TypeError: unhashable type: ’set'
```

see unhashable_type.py