Common Mistakes, Problem Solving

Dr. Carol Alexandru-Funakoshi

University of Zurich, Department of Informatics

print vs. return

- `print and return` are completely different things!
- `print`
  - Is a function that outputs a String to the console (usually)
  - Returns None
- `return`
  - Is a keyword
  - Returns a value of any type from a function (and ends the function)

```
>>> def say_my_name(name):
...   print(name)
...
>>> x = say_my_name("Bob")
Bob
>>> type(x)
<class 'NoneType'>
```

```
>>> def return_my_name(name):
...   return name
...
>>> y = return_my_name("Bob")
>>> type(y)
<class 'str'>
```

Variable scoping

- Do NOT declare variables outside of your solution function
- Variables outside the function are in global scope
- If the solution function is called more than once, its behavior will be different!
- Do **not** use `global`!

```
s = 0
def mysum(l):
    global s
    for e in l:
        s += e
    return s

assert(mysum([1,2,3]) == 6) # passes
assert(mysum([1,2,3]) == 6) # fails, 12 != 6
```

Trouble with if/elif/else

- Every `if` starts a new condition expression!
- Use `elif` to continue an expression!

```
def numbername(x):
    res = ""
    if x == 1:
        res = "one"
    if x == 2:
        res = "two"
    if x == 3:
        res = "three"
    else:
        res = "I can only count to three"
    return res


print(numbername(2))
```

Trouble with if/elif/else

- Every `if` starts a new condition expression!
- Use `elif` to continue an expression!
- Or: be smart in how you use return

```
def numbername(x):
    if x == 1:
        return "one"
    if x == 2:
        return "two"
    if x == 3:
        return "three"
    return "I can only count to three"


print(numbername(2))
```

Returning **None** by accident

- If you're supposed to return a value, make sure you actually do in all cases!

```
def index_first_even(x):
    if x != []:
        for i, n in enumerate(x):
            if n % 2 == 0:
                return i
    else:
        return -1


print(index_first_even([1,2,3])) # 1
print(index_first_even([1,3,5])) # None
```

Returning **None** by accident

- If you're supposed to return a value, make sure you actually do in all cases!

- Solutions that *look* better are usually (but not always) better
- Simple solutions are often better than complicated ones
- Think before you implement!

```
def index_first_even(x):
    for i, n in enumerate(x):
        if n % 2 == 0:
            return i
    return -1


print(index_first_even([1,2,3])) # 1
print(index_first_even([1,3,5])) # -1
```

Important! Test all input cases!!!

- A quality assurance engineer walks into a bar.
  - orders a beer
  - orders 2 beers
  - orders 0 beers
  - orders 2^{2049} beers
  - orders -1 beers
  - orders 2.75 beers
  - orders a lizard
  - orders a agh)(!^%@_05"; drop table "account";--
  - orders a `\n\t\x00`
- You can assume that only parameters that match the task description will be used in grading
- Thiscan still be a broad range of parameters and edge-cases

Review this!

- You must know how functions work in Python!
  - A function has zero or more parameters; some could be optional
  - Know how to call a function
  - Know that functions can be passed as parameters and returned
  - Global variables vs. parameters
- Know how to use lists, tuples, dicts
  - Know about enumerate, .values(), .items(), x in y, etc...
- Know about string manipulation (find(), split(), strip(), join(), etc.)

A few more things…

- Using the `global` keyword is 100% forbidden.
  - You shouldn't be specifying variables outside your solution function anyway
- Read the task carefully
  - "a list", "a tuple" or "a dictionary" implies that these could be empty. Otherwise, the task would explicitly say "a non-empty list", "a non-empty dictionary", etc...
  - Mind terms such as "non-negative", "positive", "integer", "number", etc.
- Prepare your environment!
  - IDE ready? Most important Python documentation API docs open? Lecture slides at hand? Battery full or plugged into wall? Stable internet connection?

The rules are simple

- LLM / AI: strictly forbidden
  - No ChatGPT, Claude, CoPilot, or any other LLM
  - Make 100% sure you disabled or uninstalled any LLM-assistants that may ship with your operating system, IDE, browser, or any other tool you might use
- Direct communication: strictly forbidden
  - No chat, email, message board, or any other form of communication where you actively interact with others. No collaborating. No receiving help.
- Supertab must be activated during the entire exam period
- Other than this, the exam is open-book
  - You may search Google, StackOverflow, Python docs, tutorials, blogs, etc.
  - IDE auto-completion functionality is permitted, **as long as it does not use LLM**