CONTROL, ENVIRONMENT DIAGRAMS

CS 61A

June 25, 2021

1 Control

Control is the use of boolean expressions to prevent or allow a block of code to run based on a condition. We use `if` statements and `while` loops in conjunction with these boolean expressions depending on how we want the code to behave. `if/elif/else` blocks will execute the first block of code for which the condition is `True`, whereas `while` loops will repeatedly execute a block of code while the condition is `True`.

1. To handle discussion section overflow, TAs may direct students to a more empty section that is happening at the same time.

   Write a function that takes in the number of students in two sections and prints out what to do if either section exceeds 30 students.

```
def handle_overflow(s1, s2):
    """
    >>> handle_overflow(27, 15)
    No overflow
    >>> handle_overflow(35, 29)
    Move to Section 2: 1
    >>> handle_overflow(20, 32)
    Move to Section 1: 10
    >>> handle_overflow(35, 30)
    No space left in either section
    """
```

2. Implement `pow_of_two`, which takes in an integer `n` and prints all the positive, integer powers of two less than or equal to n. This function should return `None`.

   Follow up question: What would you change about your solution if the question asked to print all the powers of two ***strictly less than*** *`n`?*

```
def pow_of_two(n):
    """
    >>> pow_of_two(6)
    1
    2
    4
    >>> result = pow_of_two(16)
    1
    2
    4
    8
    16
    >>> result is None
    True
    """
```

3. Fill out the function `min_fact` which calculates the product of consecutive positive numbers starting from `n` and working downwards until the first point at which the product becomes greater than `margin` . It should return -1 if there is no product that is greater than margin.

```
def min_fact(n, margin):
    """
    >>> min_fact(5, 20) # 5 * 4 * 3
    60
    >>> min_fact(5, 200)
    -1
    >>> min_fact(5, 0)
    5
    """
    total, ________________ = n, ________________
    while ________________:
        _____________, _____________ = _____________, _____________
    if ________________:
       ________________
    return ________________
```

2 Environment Diagrams

An **environment diagram** is a model we use to keep track of all the variables that have been defined and the values they are bound to. We will be using this tool throughout the course to understand complex programs involving several different assignments and function calls.

1. When do we make a new frame in an environment diagram?

2. Draw the environment diagram that results from running the following code.

```
def swap(x, y):
    x, y = y, x
    return print("Swapped!", x, y)

x, y = 60, 1
a = swap(x, y)
swap(a, y)
```

3. Draw the environment diagram that results from running the following code.

```
def funny(joke):
    hoax = joke + 1
    return funny(hoax)

def sad(joke):
    hoax = joke - 1
    return hoax + hoax

funny, sad = sad, funny
result = funny(sad(2))
```

3 Challenge Control Problem

1. Fill out the function `digit_div` which returns an integer that contains in any order all the digits of `k` that divide `n` evenly. If no such digit of `k` exists, the function should return 0. Assume that both `n` and `k` are positive integers.

```
def digit_div (n, k):
    >>> digit_div(4, 1234567890)
    421
    >>> digit_div(4, 2323)
    22
    >>> digit_div(7, 2323)
    0
    """
    ______________________
    while _________________________:
        curr_digit = k % 10
        if _________________________________________:
           _________________________________________
        ____________________________________________
    return _______________
```