# CS 1 – Midterm

## Week 1: Turtle and Python

* Turtle: python package to draw images on the console
* Import turtle
* Import turtle as t

|  |  |  |  |
| --- | --- | --- | --- |
| t.speed(0) | t.left(angle) | t.back(length) | t.hideturtle() |
| t.circle(radius) | t.right(angle) | t.up() | t.bye() |
| t.done() | t.forward(length) | t.clear() |  |

* Home: pen down, center of window, facing right
* Pre/post-conditions: explicitly describe the initial state of the turtle as well as where the turtle ends up your function is called
* Algorithm: series of steps (instructions) the program follows

## Week 2: Parameters, Conditionals, Fruitful Functions

* Parameter: input value for a function
* Argument: value provided when a function is called
* Input: interacting with the user
  + Always a string! Number input must be converted to int
* Variable
  + Getting input from user
  + Converting/casting type string to type int
* Conditional Statements
  + If( ):
  + Elif():
  + Else():
  + Rational Operators
    - ==, <, >, <= ,>=, !=
  + Logical Operators
    - and, or, not (or !)
* Functions:
  + def func (param, param):
  + Proper indentation necessary
  + Functions separated by newline
  + Keywords
    - Pass – leaves function
    - Break – leaves a loop
    - Return – returns the variable used within the function – can be used outside function now
    - Continue – skips rest of while loop statements
* Main Function
  + Function that’s called to run your program
  + If function isn’t called by main, then your program won’t run it
* If \_ \_name\_ \_== “main”:
  + if \_\_name\_\_ == "\_\_main\_\_": main()

## Week 3: Fruitful Functions/Recursions

* A fruitful function has fruit which means it returns a value
* Print vs. Returns
  + Print: value sent directly to console
  + Return: value can used/manipulated in other functions (outside scope of initial function)
* A recursion function is a function that calls itself
  + 2 parts:
    - Base case: ends recursion
    - Recursion case: where the function calls itself
    - Additional parameters can tell you when to stop
* Substitution Trace
  + The function (with parameters) = what each return evaluates to
  + Code sample:
    - 2 \* hanoi(4 - 1) + 1
    - 2 \* hanoi(3) + 1
    - 2 \* (2 \* hanoi(2) + 1) + 1
    - 2 \* (2 \* (2 \* hanoi(1) + 1) + 1) + 1
    - 2 \* (2 \* (2 \* 1 + 1) + 1) + 1 = 15
* Fruitful Recursion: A recursive function that returns a value

## Week 4: Tail Recursion and Iteration

* Tail Recursion has a tail which means that the last thing it does is the recursive case
* Accumulator (acc): temporary variable – stores the value of the recent recursive case
* Iteration
  + While(): or While True:

## Week 5: Strings and Files

* A string is a sequence of characters/symbols
* Each character has a position number or index that starts from 0

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| P | Y | T | H | O | N | P | Y | T | H | O | N |
| -6 | -5 | -4 | -3 | -2 | -1 | 0 | 1 | 2 | 3 | 4 | 5 |

* Strings are immutable which means they cannot be altered after declaration
* Concatenation: two variable types must be same in order to do operations with them
* Slicing
  + str[start:end] - items start through end-1
  + str[start:] - items start through the rest of the list
  + str[:end] - items from the beginning through end-1
  + str[:] - a copy of the whole list
* ASCII
  + ord() function returns the ASCII value of a character
  + chr() function returns the character associated with the ASCII value
* Time Complexity
  + O(1)
    - str[4] – constant
  + O(N)
    - For loops – linear
    - For ch in str(0:5):
      * I+= 1 (1)
      * J+=2 (1)
      * 1+1 = 2N (+N for each calculation made)
  + O(N2)
    - Exponential time
    - For loop within a for loop
* For Loops
  + For each loop (usually used for files/strings)
    - For line in file:
    - For ch in line:
    - For in range loop
    - For index in (len(word)):
* Files
  + open() – returns a file
  + close() – closes a file when done
  + strip() – strips the empty spaces at the end of the string