python™ NERD CLUB™

QUICK REFERENCE CARD - BASICS

Editor: where you write the Python programs. Example: Spyder, PyCharm, Atom, Notepad

1. Make your code readable
   - Comment using code using '#' symbol at the beginning of the line
   - Use four spaces per indentation level.
   - Keep your lines to 79 characters or fewer.
   - Use single blank lines to group parts of your program visually.

2. To print Text to console use **print() function**

```
print("welcome to Sandbox") #note: quotes have to be at start and end of text
```

3. To read user's input from console, use **input() function**

```
name = input("please enter your name") #note: single '=' sign means assignment
#note: This is read by the computer as a String
```

4. To convert user's input from console from String to Integer, use **int() function**

```
age = int(input("what's your age?") #note:
```

5. Variable store data of different types and the data can be changed during the course of a program

```
name = input("what's your name?")
age = int(input("what's your age in years?"))
weight = float(input("what's your weight in pounds?"))

print(type(name)) #note: <class 'str'>
print(type(age)) #note: <class 'int'>
print(type(weight)) #note: <class 'float'>
```

6. Program is a set of commands that are executed from top to bottom. To execute certain commands based on certain conditions, use **if –elif-else** statements

```
if name == "Bob":   #note: double '=' sign means comparison
        print("welcome", name)
elif name == "Jane":
        print("you are wonderful", name)
else: #note: indentation, and colon to indicate start of paragraph
   print("go home", name)
   #note: the comma indicates concatenation, and 'name' is the name of the variable
```

7. Loops are used to repeat commands: While and For loops are the 2 types of Loops

```
# sample program below prints a number series using while loop
print("this program prints a natural number series")
limit = input("select your upper limit for the series")
i = 1
while i <= limit:
#note: commands inside while loop are executed repeatedly until the condition becomes false
        print(i)
        i = i + 1
print("the number series has ended")
```

8. Loops are used to repeat commands: While and For loops are the 2 types of Loops

```
# sample program below prints a number series using for loop
print("this program prints a natural number series")
limit = input("select your upper limit for the series")
i = 1
for i  in range(1, limit, 1):  #range() function includes 3 arguments: start, end, increment
#note: commands inside for loop are executed repeated until the condition becomes false
        print(i)
print("the number series has ended")
```

9. There are 2 types of Functions: Built-In and User-Defined.
   print() is a built-in function.
   The function milesToKm() is a user-defined function

```
def milesToKm(miles): #note: def is the keyword used to define a function
        km=miles*1.61
        return km #note: this function when executed returns a value, km

print(milesToKm(10)) #note: here the function is being called
```

10. Libraries allow your Python code to access additional functions. For example, create random numbers.

```
import time #note: this library allows access to delay, time, calendar, etc.
import random #note: this l

randNum = random.randint(1,10) #note: this generates a random number between 1 and 10
print("I am going to sleep for 3 seconds")
time.sleep(3)
print('I am awake now')
```

STRINGS:

11. Creating Strings: multiple ways to create strings

```
#note: all of the following are equivalent
my_string = 'Hello'
print(my_string)

my_string = "Hello"
print(my_string)

my_string = '''Hello'''
print(my_string)

# triple quotes string can extend multiple lines
my_string = """Hello, welcome to
        the world of Python"""
print(my_string)
```

12. You can concatenate strings

```
#concatenate using either comma or plus sign
#comma automatically adds a plus sign and an empty space
print('hello',"world")
print('hello'+"world")
print('hello', 5) # prints hello 5
print('hello'+5) # throws an error, as it connect convert 5 to a string to perform the '+' operation
```

13. Strings have indexes & you can calculate string length

```
str = "hEllo wORld"
print(str)
print(len(str)) #prints the length of the string, which is 11 in this case, and includes 1 space

print(str[1]) #prints the 2nd character of the str, as indexing starts from 0

print(str.upper())
print(str.lower())
```

14. Accepting user input and manipulating it

```
name = raw_input("What's your name? ")
print("Hello, " + name + ".")
```

15. Python allows method chaining

```
#Python uses ZERO-BASED INDEXING
# Python is CASE-SENSITIVE
# Python allows METHOD CHAINING

print("index of letter 'r' in the string is %d" %str.lower().index('r'))
#Python cannot find 'r' until the entire string is converted to lower case
```

16. Split a string

```
name = "Bryan Cairns"
mylist = name.split(" ")
print(mylist[0]+mylist[1])
print(name) #original variable was not modified by the split() function
```

17. Replace a character in a string

```
name1 = "Apple pie"
name2 = name1.replace('p','x') #replaces all instances of 'p'
```

18. replace white/empty space in a string

```
c = "    apple pie     "
print(c.strip())
```

19. Sample Program: check if a string is a palindrome

```
myS = input("please enter your string:")
newS=""

i = len(myS)
j = 0

while (i > 0):
   newS=newS+myS[i-1]
   # creates a new string with each iteration, and stores the reference to that new string in newS
   #newS[0]=myS[i] gives an error: 'str' object does not support item assignment
   i = i-1
   j=j+1

print("the reserve of the string is:",newS)

if newS == myS:
   print("this is a palindrome")
else:
   print("sorry, not a palindrome")
```

