Python Reference Card

1. Programs and Functions

The following is an example of a Program (Calculator) with a single "main" section:

```
if __name__ == '__main__':
    print("Calculator")
```

The following is an example of a function declaration:

```
def circle_area(radius):
    area = math.pi * radius ** 2
    return area
```

The following is an example of an invocation of this function:

```
radius = 5.0
area = circle_area(radius)
```

3. Operators

Arithmetic	Operators
Addition	+
Division	/
Int/floor Division	//
Multiplication	*
Modulus	%
Negation	-
Subtraction	-

4. Type Conversion

Example Expression	Type	Value
(1 + 2 + 3 + 4)/4	float	2.5
"1234" + str(99)	str	"123499"
11 * 0.25	float	2.75
int(2.71828)	int	2
int(11) * 0.25	float	2.75
11 * int(0.25)	int	0
int(11 * 0.25)	int	2

5. Math Library Methods/Constants

Signature	Purpose	Return type
`math.fabs(v)`	Absolute value	float
`math.cos(a)`	Cosine of angle a in radians	float
`math.pow(v, p)`	`v` raised to the `p` power	float
`math.pi`	The constant for p	N/A

6. Python functions

`max(obj)` → returns the largest of an iterable (string, list, tuple etc.)
`min(obj)` → returns the smallest of an iterable (string,list,tuple,etc)
`len(obj)` → returns the length of an iterable (string, list, tuple etc.)
`sum()` → sums the items of an iterable (list, tuple, etc.)

7. Lists

```
my_list = ["item1", "item2", "item3"]
```

```
my_list.append(val)     → adds an element (val) to a list
my_list.pop()           → removes the last element and returns the value
my_list.pop(i)          → removes an element at at a given index (i)
my_list.copy()          → returns a copy of the list
my_list.count(val)      → returns the number of occurrences of a value
my_list.insert(i, val)  → adds an element at a specified position (i)
my_list.remove(val)     → removes the first element with the specified value
my_list.reverse()       → reverses the order of the list
my_list.sort()          → sorts the list in ascending order
```

8. Input

Input command

```
d = float(input("Optional Prompt"))

i = int(input())

s = str(input())
```

9. Output

`print()`	Can be passed a `float`, `int`, or `str`
`print("hello", end=" ")`	Use of the named argument `end` to change what is printed at the end of a line (default is `end="\n"`)
`print("hello", "world", sep="_")`	Use of the named argument `sep` to change what is printed in between values (default is `end=" "`)
`print(f"my name is {name}")`	Is passed an f-string with a variable referenced in the {}
`print(f"this {variable:.2f}")`	Is passed an f-string with a variable and a format specifier .2f

Example Specifier	**Description**
`s`	String
`d`	Integer
`f`	Fixed-point notation with 6 places of precision (for floats)
`.`**`[precision]`**`f`	Fixed-point notation with **[precision]** places of precision
`[width].[precision]``f`	Fixed-point notation with precision and right aligned to **[width]**
`0`**`[precision]`**`d`	Leading 0 notation (prepends integers with **[precision]** zeros)
`>`**`[width]`**`s`	Right-align text with spaces to **[width]** number of characters

Complete Example

```
                                  12345678901234567890
print(f"{5:3}{8:6.2f}")             5  8.00
print(f"{'hi':>5s} {1:05d}")        hi 00001
```