

6.1 Quick Reference Guide

The following is a quick reference guide for H2 Computing that students can use as a reference when attempting practical questions to reduce memory load.

1. Python

1. Identifiers

When naming variables, functions and modules, the following rules must be observed:

- Names should begin with character 'a' - 'z' or 'A' - 'Z' or '_' and followed by alphanumeric characters or '_'.
- Reserved words should not be used.
- User-defined identifiers are case sensitive.

2. Comments and Documentation Strings

```
# This is a comment
```

```
"""
    This is a documentation string
    over multiple lines
"""
```

3. Input/Output

```
print ("This is a string")
```

```
s = input ("Instructions to prompt for data entry.")
```

4. Import

```
import <module>
```

```
from <module> import <name>
```

5. Data Type

Data Type	Notes
int	integer
float	real number
bool	boolean
str	string (immutable)
list	series of values
dict	key-value pairs
tuple	series of values (immutable)

6. Assignment

Assignment Statement	Notes
a = 1	integer
b = c	variable
d = "This is a string"	string
mylist = [1, 2, 3, 4, 5]	list
mydict = {'key': 'value'}	dict

7. Arithmetic Operators

Operator	Notes
+ -	plus, subtract
* /	multiply, divide
%	remainder or modulus
**	exponential or power
//	quotient of the floor division

8. Relational Operators

Operator	Notes
==	equality
!=	not equal to
> >=	greater than, greater than or equal to
< <=	less than, less than or equal to

9. Boolean Expression

Boolean Expression	Notes
a and b	logical and
a or b	logical or
not a	logical not

10. Iteration

while loop	for loop
while condition(s): <statement(s)>	for i in range(n): <statement(s)>
	for record in records: <statement(s)>

11. Selection

Type 1	Type 2	Type 3
if condition(s): <statement(s)>	if condition(s): <statement(s)> else: <statement(s)>	if condition(s): <statement(s)> elif condition(s): <statement(s)> else: <statement(s)>

12. Functions

Function definitions

@<optional decorator(s)>

def <function name> (<parameters>):
 <function body>

Function calls

<function name>(<value>, <name>=<value>)

13. Object-Oriented Programming

class <class name> (<optional parent class>):

def __init__(self, <parameters>):
 <constructor body>

def <method name> (self, <parameters>):
 <method body>

14. Built-in Functions and Attributes

__file__	<file>.readlines()	<list>.copy()	print()	<str>.isdigit()
__name__	<file>.write()	<list>.index()	range()	<str>.islower()
abs()	float()	<list>.insert()	round()	<str>.isspace()
bin()	hex()	<list>.pop()	staticmethod()	<str>.isupper()
<bytes>.decode()	input()	<list>.remove()	str()	<str>.lower()
chr()	int()	<list>.reverse()	<str>.encode()	<str>.startswith()
<dict>.clear()	len()	<list>.sort()	<str>.endswith()	<str>.upper()
<dict>.copy()	list()	max()	<str>.format()	
<file>.close()	<list>.append()	min()	<str>.index()	
<file>.read()	<list>.extend()	open()	<str>.isalnum()	
<file>.readline()	<list>.clear()	ord()	<str>.isalpha()	

csv module	datetime module	math module
reader() writer() <writer>.writerow()	datetime() datetime.now() datetime.strptime() <datetime>.isoformat() <datetime>.strftime() <datetime>.year <datetime>.month	<datetime>.day <datetime>.hour <datetime>.minute <datetime>.second <timedelta>.days <timedelta>.seconds ceil() exp() floor() log() pow() sqrt() trunc()

os.path module	random module	sqlite3 module	socket module	sys module
basename() dirname() isdir() isfile() join()	random() randint() randrange() shuffle()	connect() <connection>.commit() <connection>.close() <connection>.execute() <connection>.rollback() <connection>.row_factory <cursor>.fetchone() <cursor>.fetchall() Row	socket() bind() listen() accept() connect() recv() sendall()	exit()

15. Additional Functions and Attributes

pymongo module		flask module
MongoClient()	<collection>.update_one()	Flask()
<client>.database_names()	<collection>.update_many()	<flask application>.route()
<client>.get_database()	<collection>.delete_one()	<flask application>.run()
<client>.drop_database()	<collection>.delete_many()	render_template()
<client>.close()	<collection>.count()	request.files
<database>.collection_names()	<cursor>.count()	request.form
<database>.get_collection()		request.method
<database>.drop_collection()		send_from_directory()
<collection>.insert_one()		redirect()
<collection>.insert_many()		url_for()
<collection>.find_one()		secure_filename()
<collection>.find()		<uploaded file>.save()

2. SQL Statements

CREATE TABLE <i>table_name</i> (<i>column1_name COLUMN1_TYPE COLUMN1_CONSTRAINTS</i> , <i>column2_name COLUMN2_TYPE COLUMN2_CONSTRAINTS</i> , ... PRIMARY KEY (<i>column1_name, column2_name, ...</i>), FOREIGN KEY (<i>column_name</i>) REFERENCES <i>table_name</i> (<i>column_name</i>));	
SELECT <i>column1_name, column2_name, ...</i> FROM <i>table_name</i> WHERE <i>where_expression</i> ORDER BY <i>order_expression</i> ASC ;	SELECT <i>column1_name, column2_name, ...</i> FROM <i>table_name</i> WHERE <i>where_expression</i> ORDER BY <i>order_expression</i> DESC ;
SELECT <i>table1_name.column1_name, table2_name.column2_name, ...</i> FROM <i>table_name, table2_name</i> WHERE <i>where_expression</i> ;	
SELECT <i>table1_name.column1_name, table2_name.column2_name, ...</i> FROM <i>table1_name</i> INNER JOIN <i>table2_name</i> ON <i>join_expression</i> ;	
SELECT <i>table1_name.column1_name, table2_name.column2_name, ...</i> FROM <i>table1_name</i> LEFT OUTER JOIN <i>table2_name</i> ON <i>join_expression</i> ;	
SELECT <i>COUNT(*),</i> <i>MAX(column1_name),</i> <i>MIN(column2_name),</i> <i>SUM(column3_name),</i> ... FROM <i>table_name</i> ;	

```
INSERT INTO table_name(column1_name, column2_name, ...)
VALUES(column1_value, column2_value, ...);
```

```
UPDATE table_name SET
    column1_name = column1_expression,
    column2_name = column2_expression,
    ...
WHERE where_expression;
```

```
DELETE FROM table_name
WHERE where_expression;
```

```
DROP TABLE table_name;
```

3. SQLite Types, Constraints, Functions and Operators

Types	Constraints	Functions	Operators			
NULL	NOT NULL	COUNT()		/	<	AND
REAL	PRIMARY KEY	MAX()	+	%	<=	OR
INTEGER	AUTOINCREMENT	MIN()	-	=	>	IS
TEXT	UNIQUE	SUM()	*	!=	>=	IS NOT

4. PyMongo Operators

Comparison

\$eq	\$gt	\$gte	\$lt	\$lte
\$ne	\$in	\$nin		

Logical

\$and	\$not	\$or
-------	-------	------

Element

\$exists

Update

\$set	\$unset
-------	---------

5. HTML Elements, Attributes and Character References

The first line of a HTML document must be: <!doctype html>

Type	Elements	Attributes
Common		id, class
Required	<html>, <head>, <title>, <body>	
Metadata	<link>	rel, href
Structure	<h1>, <h2>, <h3>, <p>, <div>, , <hr>	
Text and Media	, <i>	
	<a>	href
		src, alt
Table	<table>, <tr>, <th>, <td>	
Form	<form>	action, enctype, method
	<input>	name, type, value
	<textarea>	name

Character	&	<	>	"
Reference	&amp;	&lt;	&gt;	&quot;

6. Jinja2 Filters

length	safe
--------	------

7. CSS Properties

Common	Box Model		Typography
display background color	height width border border-bottom border-left border-right border-top margin margin-bottom	margin-left margin-right margin-top padding padding-bottom padding-left padding-right padding-top	font-family font-size font-style font-weight text-align text-decoration