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

```
if condition(s):
    <statement(s)>
```

Type 2

```
if condition(s):
    <statement(s)>
else:
    <statement(s)>
```

Type 3

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

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

15. Additional Functions and Attributes

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

2. SQL Statements

CREATE TABLE *table_name*(
column1_name COLUMN1_TYPE COLUMN1_CONSTRAINTS,
column2_name COLUMN2_TYPE COLUMN2_CONSTRAINTS,
...
PRIMARY KEY (*column1_name*, *column2_name*, …),
FOREIGN KEY (*column_name)* **REFERENCES** *table_name(column_name)*
);

SELECT *column1_name*, *column2_name*, ...
FROM *table_name*
WHERE *where_expression*
ORDER BY *order_expression* **ASC**;

SELECT *column1_name*, *column2_name*, ...
FROM *table_name*
WHERE *where_expression*
ORDER BY *order_expression* **DESC**;

SELECT *table1_name.column1_name*, *table2_name.column2_name*, ...
FROM *table_name*, *table2_name*
WHERE *where_expression*;

SELECT *table1_name.column1_name*, *table2_name.column2_name*, ...
FROM *table1_name*
INNER JOIN *table2_name* **ON** *join_expression;*

SELECT *table1_name.column1_name*, *table2_name.column2_name*, ...
FROM *table1_name*
LEFT OUTER JOIN *table2_name* **ON** *join_expression;*

SELECT
COUNT(*),
MAX(column1_name),
MIN(column2_name),
SUM(column3_name),
...
FROM *table_name;*

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>, , , <hr>	
Text and Media	<b>, <i>	
	<a>	href
	<img>	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