**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.

## **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() | 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() | <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() | | |

## **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*; | |

## **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 |

## **PyMongo Operators**

**Comparison**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| $eq | $gt | $gte | $lt | $lte |
| $ne | $in | $nin |  |  |

**Logical Element**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| $and | $not | $or |  | $exists |

**Update**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| $set | $unset |  |  |  |

## **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>, <span>, <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; |

## **Jinja2 Filters**

|  |  |
| --- | --- |
| length | safe |

## **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 |