H2

MINISTRY OF EDUCATION, SINGAPORE
in collaboration with
CAMBRIDGE INTERNATIONAL EXAMINATIONS
General Certificate of Education Advanced Level

Reference Guide

for

Computing (Advanced Level)

For use from 2027 in Paper 2 for the H2 syllabus.

This document has **8** pages.

Singapore Examinations and Assessment Board

CAMBRIDGE
International Education

1 Python

1.1 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.

1.2 Comments and Documentation Strings

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

1.3 Input/Output

```
s = input("Prompt for data: ")

print("This is a string")

f = open("input.txt", "r")
line = f.readline()
character = f.read(1)
f.close()

with open("output.txt", "w") as f:
    f.write("Output Line\n")
```

1.4 Import

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

1.5 Data Types

Type	Example	Notes
`int`	`-3`	integer
`float`	`3.1415926`	real number
`bool`	`True`	Boolean
`str`	`"Hello"`	string (immutable)
`list`	`[2, 3, 5]`	series of values
`dict`	`{'key':'value'}`	key-value pairs
`tuple`	`(2, 3, 5)`	series of values (immutable)

1.6 Assignment

Statement	Notes
`a = 1`	normal assignment
`b += c`	augmented assignment equivalent to `b = b + c`
`x[y] = z`	assigns z to index y of list x or assigns z to key y of dictionary x
`del a`	deletes variable a
`del x[y]`	deletes key y and its value from dictionary x

1.7 Arithmetic Operators

Operator	Notes
`+ -`	add, subtract
`* /`	multiply, divide
`%`	remainder or modulo
`**`	exponential or power
`//`	floor division

1.8 Relational Operators

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

1.9 Boolean Expression

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

1.10 Sequence (List/String) Operations

Operator	Notes
`<seq> + <seq>`	concatenation
`<int> * <seq>`	repetition
`<seq>[index]`	indexing
`<seq>[start:stop]`	slicing
`<seq>[start:stop:skip]`	slicing with skip
`<value> in <seq>`	membership testing

1.11 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)>
```

1.12 Iteration

while loop

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

for loop

```
for i in range(n):
    <statement(s)>

for record in records:
    <statement(s)>
```

1.13 Functions

```
# Function definition
@<optional decorator(s)>
def <function name>(<parameters>):
    <function body>
    return <return value>

# Function call
<function name>(<value>, <name>=<value>)
```

1.14 Object-Oriented Programming

```
# Class definition
class <class name>(<optional parent class>):

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

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

1.15 Built-in Functions and Attributes

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

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()	pow()
	<datetime>.year	<timedelta>.days	sqrt()
	<datetime>.month	<timedelta>.seconds	trunc()

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

1.16 Additional Functions and Attributes

sklearn.neighbors module	sklearn.cluster module	sklearn.model_selection module
KNeighborsClassifier()	KMeans()	cross_validate()
<classifier>.classes_	<estimator>.cluster_centers_	train_test_split()
<classifier>.n_features_in_	<estimator>.labels_	
<classifier>.n_samples_fit_	<estimator>.inertia_	
<classifier>.fit()	<estimator>.n_iter_	
<classifier>.get_params()	<estimator>.n_features_in_	
<classifier>.kneighbors()	<estimator>.fit()	
<classifier>.predict()	<estimator>.fit_predict()	
<classifier>.predict_proba()	<estimator>.get_params()	
<classifier>.score()	<estimator>.predict()	
<classifier>.set_params()	<estimator>.score()	
	<estimator>.set_params()	

numpy module		flask module
`amax()`	`linspace()`	`Flask()`
`amin()`	`log()`	`<flask application>.route()`
`append()`	`mean()`	`<flask application>.run()`
`arange()`	`ones()`	`render_template()`
`array()`	`power()`	`request.files`
`<array>.copy()`	`reshape()`	`request.form`
`ceil()`	`resize()`	`request.method`
`concatenate()`	`round()`	`send_from_directory()`
`cumsum()`	`shape()`	`redirect()`
`delete()`	`sqrt()`	`url_for()`
`dot()`	`std()`	`secure_filename()`
`exp()`	`sum()`	`<uploaded file>.save()`
`empty()`	`transpose()`	
`floor()`	`trunc()`	
`insert()`	`zeros()`	

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 table1_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	IS
REAL	PRIMARY KEY	MAX()	+	%	<=	OR	IS NOT
INTEGER	AUTOINCREMENT	MIN()	-	=	>	NOT	
TEXT	UNIQUE	SUM()	*	!=	>=		

4 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;

5 Jinja2 Filters

length	safe

6 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