Grade 8 Computing

Unit 3 · Algorithms: Searching and Sorting

ANSWER KEY · FOR TEACHERS

Full marks 34

Give the marks for the points in the marking scheme; an answer in other words that says the same is right.

1 Key words

1. flowchart
2. index
3. Linear search
4. Binary search
5. sorted
6. comparison
7. pass
8. edge

Each carries 0.5 mark. **K · 4**

2 Label the diagram

1. Start or stop where the algorithm begins and ends
2. Procedure a job that has a flowchart of its own
3. Process a step that does something: set or work out a value
4. Input or output ask for a value, or show one
5. Connector joins two parts without a long arrow
6. Decision a question with two ways out: yes or no

Each carries 0.5 mark. **K · 3**

3 Multiple choice

i. **c.** 19 **U · 1**

`len(roll)` is 5, so this is `roll[4]`, the last item: 19. `roll[5]` would be an IndexError.

ii. **c.** 14, 3, 27, 8, 19 **U · 1**

14, 3, 27, 8, 19 is not in order, so binary search could throw away the half that holds the answer. Sort it first. The other three are sorted.

iii. **d.** 6 **A · 1**

With 6 looks you can search 1 + 2 + 4 + 8 + 16 + 32 = 63 items: 63 → 31 → 15 → 7 → 3 → 1 → 0. Linear search could need all 63.

iv. **b.** the biggest, at the right end **U · 1**

Each comparison carries the bigger of two neighbors one place to the right, so the biggest item is carried all the way to the end in the first pass. The smallest may move only one place left in each pass.

v. **b.** 40 **U · 1**

40 is the largest roll number allowed, at the very edge. 17 is ordinary; 52 and "twelve" are wrong inputs, which the program should refuse sensibly.

4 What does it print?

MARKING SCHEME **U · 3**

- ✓ `52 60 5`: positions start at 0, so position 4 is the last of five items.
- ✓ `45`: `len(marks) - 2` is 3, the second-to-last position.
- ✓ `True False`: 38 was changed to 40 on line 2, so it is no longer in the list.

Science Playground, Katunje, Bhaktapur
Book a class visit: https://science-playground.com/booking

5 Trace a binary search

ANSWER **A · 4**

Look	low	high	middle	Number there	So
1	0	8	4	25	bigger than 20: high = 3
2	0	3	1	9	smaller: low = 2
3	2	3	2	13	smaller: low = 3
4	3	3	3	18	smaller: low = 4

Now low is 4 and high is 3: low has passed high, so no positions are left where 20 could be, and the search says it is not in the list, after 4 comparisons.

MARKING SCHEME

- ✓ Look 2: middle 1, the number 9, low becomes 2.
- ✓ Look 3: middle 2, the number 13, low becomes 3.
- ✓ Look 4: middle 3, the number 18, low becomes 4.
- ✓ low (4) has gone past high (3), so nothing is left to search: 20 is not in the list.

6 Sort by hand

MARKING SCHEME **A · 3**

- ✓ The lists after passes 1 and 2: 14, 29, 8, 35 and 14, 8, 29, 35.
- ✓ Pass 3 gives 8, 14, 29, 35, and a fourth pass with no swaps ends the sort.
- ✓ 12 comparisons and 4 swaps.

7 Find and fix the bug

MARKING SCHEME **A · 3**

- ✓ Line 5: <= lets **i** reach 5, a position that doesn't exist (positions 0 to 4).
- ✓ `while i < len(names) and not found:`.
- ✓ A name that is found stops the loop early; only a name not in the list goes past the end.

Science Playground, Katunje, Bhaktapur
Book a class visit: https://science-playground.com/booking

8 Write a program

ANSWER **A · 4**

```
names = ["Sita", "Bikash", "Gita", "Hari", "Maya", "Nabin"]
target = input("Whose name? ")
i = 0
found = False
while i < len(names) and not found:
    if names[i] == target:
        found = True
    else:
        i = i + 1
if found:
    print(target, "is at position", i)
else:
    print(target, "is not in the list.")
```

```
OUTPUT
Whose name? Hari
Hari is at position 3
```

A `for i in range(len(names))`: loop that prints the position and sets a flag when the name is found is right too, as long as "not in the list" is printed once, after the loop, and only when the name wasn't found.

MARKING SCHEME

- ✓ The target asked with `input()`, and a loop over the positions, 0 to `len(names) - 1`.
- ✓ Each item compared with the target (`names[i] == target`).
- ✓ A flag, or a stop, when it is found, so the position is known.
- ✓ After the loop: the position, or "not in the list", printed once.

9 Choosing tests

MARKING SCHEME **A · 3**

- ✓ An ordinary test, such as 17, expecting that student's name.
- ✓ An edge test, 1 or 40 (the first or the last), expecting that student's name.
- ✓ A wrong test, such as 0, 41 or "twelve", expecting a sensible message instead of a crash.

10 Think

MARKING SCHEME **HA · 2**

- ✓ Binary search works only on a sorted list, and sorting takes much more work than one linear search.
- ✓ So for a short list, a list searched only once, or one that changes all the time, linear search is the better choice; a list searched again and again is worth keeping sorted.

11 Activity: Sort like a computer

(not marked)

WHAT STUDENTS SHOULD FIND

For 8 well-shuffled slips, bubble sort usually needs 35 to 56 comparisons (always a multiple of 7, because each pass compares 7 pairs), and insertion sort usually 15 to 25. Sorting with only two numbers visible at a time needs a plan, and a plan is an algorithm.

Marks by level	Knowledge	Understanding	Application	Higher ability	Total
Marks	7	7	18	2	**34**

Science Playground, Katunje, Bhaktapur
Book a class visit: https://science-playground.com/booking