Unit 3 · Data Structures

WORKSHEET

Full marks 33

Name _____ Class _____ Roll no. ____ Date _____ Marks _____ / 33

1 Key words [4]

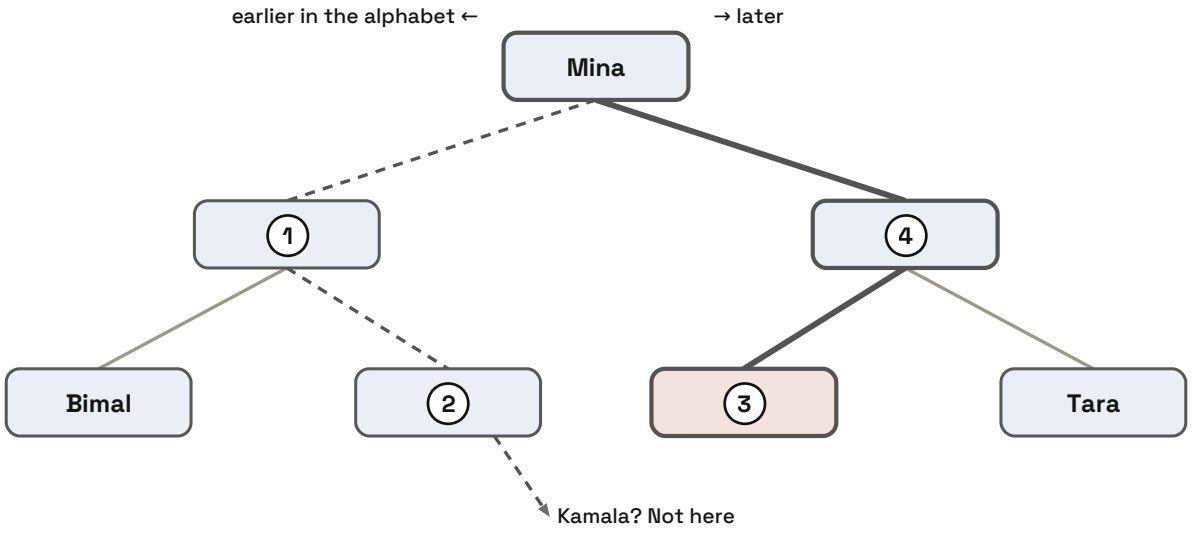
Fill in the blanks with words from the box. Two words in the box are not needed. Each blank carries 0.5 mark.

binary search tree	**collision**	**graph**	**hash table**	**heap**	**linked list**	**node**	**queue**	**stack**
tuple								

A _____ takes away the item added last, and a _____ the item added first. In a _____ each _____ holds a value and a reference to the next one. A _____ works out a key's bucket from the key itself; two keys sent to the same bucket are a _____. In a _____ every key on a node's left is smaller than the node's key, and every key on its right bigger. A _____ always keeps its smallest item at the front, ready to be taken first.

2 Label the diagram [2]

The diagram is a binary search tree of names, in alphabetical order. Four names are missing: Dawa, Hari, Pemba and Sita. Write each in its place, 1 to 4. Each carries 0.5 mark.



1. _____ 2. _____

3. _____ 4. _____

3 Multiple choice [5]

Choose the correct option for each of the following questions.

i. Push 3, push 8, pop, push 5, push 1, pop, pop. What is on the top of the stack now? [1]

a. 3 **b.** 8

c. 5 **d.** 1

ii. You open pages A, B, C and D in a browser, press Back twice, then open page E. What does pressing Forward do now? [1]

a. goes to C

b. goes to D

c. nothing: the forward stack was emptied when E was opened

d. goes back to A

iii. A hash table has 10 buckets and holds 7 keys. The hash function sends every key to bucket 3. How many keys may a look-up have to compare? [1]

a. 1 **b.** 3

c. 7 **d.** 10

iv. Roll numbers 10, 20, 30, 40 and 50 are added to an empty BST in that order. How many looks does it take to find 50? [1]

a. 1 **b.** 3

c. 5 **d.** 6

v. A heap holds 1,000 patients. About how many swaps can adding one more patient need, at most? [1]

a. 1 **b.** about 10

c. about 500 **d.** 1,000

4 What does it print? [3]

Write exactly what the program prints, in the box under it.

```
1  from collections import deque
2
3  stack = []
4  queue = deque()
5  for city in ["Pokhara", "Butwal", "Dharan", "Hetauda"]:
6      stack.append(city)
7      queue.append(city)
8  print(stack.pop(), queue.popleft())
9  stack.append("Janakpur")
10 queue.append("Janakpur")
11 print(stack.pop(), stack.pop(), queue.popleft())
12 print(len(stack), list(queue))
```

```
OUTPUT
```

5 Trace the program [4]

The function checks the brackets of a text with a stack. Fill in a row for each bracket of `{[(x)]}(`, then what the program prints. The first row is filled in.

```
1  PAIRS = {")": "(", "]": "[", "}": "{"}
2
3  def balanced(text):
4      stack = []
5      for ch in text:
6          if ch in "([{":
7              stack.append(ch)
8          elif ch in PAIRS:
9              if not stack or stack.pop() != PAIRS[ch]:
10                 return False
11     return len(stack) == 0
12
13 print(balanced("{[(x)]}("))
```

ch	stack after it	Output
{	['{']	

6 Grow a search tree [4]

Answer in the box.

The keys 40, 25, 60, 10, 30, 50, 70, 27 and 55 are added, in that order, to an empty binary search tree.

i. Draw the tree.
ii. Write the keys in the order an in-order traversal visits them.
iii. How many looks does it take to find 27? How many to find that 35 isn't there?

Answer: _____

7 Find and fix the bug [3]

Read the program, then answer.

`find` should give back the node holding **target**, or None if no node holds it. The first search works, but the second stops the program with an error.

```
1  class Node:
2      def __init__(self, value, next):
3          self.value = value
4          self.next = next
5
6  def find(head, target):
7      node = head
8      while node.value != target:
9          node = node.next
10     return node
11
12 route = Node("Lagankhel", Node("Thapathali", Node("Ratnapark", None)))
13 print(find(route, "Ratnapark").value)
14 print(find(route, "Kalanki"))
```

i. Which error stops the program, and on which line?
ii. Why does it happen?
iii. Write the corrected line.

8 Short answer [2]

Write a short answer to the following question.

Choose a data structure for each job, and give a reason: i. an emergency ward that always treats the most urgent patient next, while new patients keep arriving; ii. a phone book that must find a name quickly and also list every name in alphabetical order.

9 Write a program [4]

Write the program in Python on the lines. Any program that gives the right result gets the marks.

The nodes of a binary search tree are objects of this class:

```
class Node:
    def __init__(self, key):
        self.key = key
        self.left = None
        self.right = None
```

Write two recursive functions: `height(node)`, the number of levels in the tree below and including **node** (0 for None), and `leaves(node)`, the number of nodes with no children. For the tree of **Grow a search tree** they give 4 and 4.

10 Think [2]

Think, then answer.

The same flood fill is run on a picture twice, keeping the squares still to fill on a stack and then on a queue. Both color the same 160 squares, but the stack's list grows to 248 squares, while the queue's never holds more than 44. Why do both color the same squares, and why does the stack's list grow so much longer?

11 Activity: Grow a search tree (not marked)

At a computer, open the unit's page (the QR code) and find **Grow a search tree.** Add the roll numbers in the order shown, then in sorted order, and fill in the table.

Order added	Levels	Looks to find the biggest	Looks to find my number
The order shown			
Sorted order			