Unit 4 · Algorithms and Efficiency

WORKSHEET

Full marks 30

Name _____ Class _____ Roll no. ____ Date _____ Marks _____ / 30

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.

Big-O notation	**constant**	**deterministic**	**efficient**	**exponential**	**linear**	**logarithmic**
merging	**pivot**	**quadratic**				

An algorithm that does little work, in steps and memory, for its job is _____. Linear search is _____: twice the items, twice the looks. Binary search is _____: twice the items, just one look more. Comparing every pair of items is _____, and trying every group of items is _____. Writing $O(n^2)$ for the growth is called _____. Putting two sorted piles together into one sorted pile is called _____. An algorithm that takes the same steps and gives the same output every time for the same input is _____.

2 Multiple choice [5]

Choose the correct option for each of the following questions.

i. Binary search finds a name among 1,000 sorted names in at most 10 looks. At most how many looks for 4,000 names? [1]

a. 11 **b.** 12
c. 40 **d.** 4,000

ii. A program has a loop inside a loop, and both go through the whole list of n items. How does its work most likely grow? [1]

a. $O(1)$ **b.** $O(n)$
c. $O(n^2)$ **d.** $O(2^n)$

iii. How many different groups (including taking nothing and taking all) can be made from 10 things? [1]

a. 20 **b.** 100
c. 1,024 **d.** 3,628,800

iv. A program takes 2 seconds for 1,000 items and its algorithm is quadratic. About how long will it take for 10,000 items? [1]

a. 2 seconds **b.** 20 seconds
c. 200 seconds **d.** 2,000 seconds

v. Random quicksort is run three times on the same list. What can change from run to run? [1]

a. the sorted list it gives **b.** the number of comparisons it makes
c. both **d.** neither

3 What does it print? [3]

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

```
def count_pairs(n):
    steps = 0
    for i in range(n):
        for j in range(i + 1, n):
            steps = steps + 1
    return steps

for n in [4, 8, 16]:
    print(n, count_pairs(n))
```

OUTPUT

What kind of growth is this, and why?

4 Short answers [6]

Write short answers to the following questions.

a. Thirty teams play in a league where every team plays every other team once. How many matches are there? How many if the league grows to 60 teams? [2]

b. A program that uses an $O(n^2)$ algorithm takes 3 seconds for 10,000 records. About how long will it take for 30,000 records? And if it used an $O(n)$ algorithm that also took 3 seconds for 10,000? [2]

c. What is the difference between a deterministic and a random algorithm? Give an example of each. [2]

5 Trace the program [4]

Fill in one row for each time the loop's body runs, then what the program prints. The first row shows the values before the loop.

```
1  def merge(a, b):
2      c = []
3      i = 0
4      j = 0
5      while i < len(a) and j < len(b):
6          if a[i] <= b[j]:
7              c.append(a[i])
8              i = i + 1
9          else:
10             c.append(b[j])
11             j = j + 1
12     c = c + a[i:] + b[j:]
13     return c
14
15 print(merge([12, 30, 45, 66], [8, 31, 50]))
```

a[i]	b[j]	**Added to** c	i	j
(before)			0	0

OUTPUT

6 Find the bug [2]

Answer about the program in **Trace the program.**

If line 12 is deleted, what does the program print? Why?

7 Long answer [4]

Write a long answer to the following question.

Write an algorithm, as numbered steps from Start to Stop, to find whether a list of n numbers is in order from smallest to biggest. How many comparisons does it make at most? What kind of growth is that?

8 Think [2]

Think, then answer.

A school buys a computer 10 times faster. Its program uses a quadratic sort. How much bigger can the list now be, if the sort is to take no longer than before?

9 Activity: Watch the sorts (not marked)

At a computer, open the unit's page (the QR code) and find **Watch merge sort.** Sort each list with each sort, and write how many comparisons it made.

List	Merge sort	Shorter passes	Insertion sort
Mixed			
Nearly in order			
Backwards			