Name: GOH WEI ZHE

Total: 20 marks

6

1. Let A= [9, 17, 3, 5, 10, 20, 2]. Write down the values of A for each pass, when you sort it using bubble sort. (5 marks)

A = [9, 17, 3, 5, 10, 20, 2]

~~A = [9, 3, 17~~

A = [9, 3, 5, 10, 17, 2, 20]

A = [3, 5, 9, 10, 2, 17, 20]

A = [3, 5, 9, 2, 10, 17, 20]

A = [3, 5, 2, 9, 10, 17, 20]

A = [3, 2, 5, 9, 10, 17, 20]

A = [2, 3, 5, 9, 10, 17, 20]

5

2. Consider the following lists of partially sorted numbers. The bar represents the sort marker. How many comparisons and shifts are needed to sort the next number using the insertion sort [2, 3, 39, 90, 95, | 25, 30]. (2 marks)

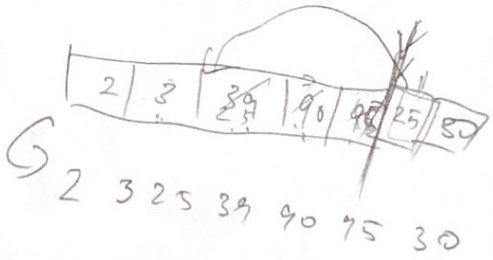
7 marks

2, 3, 25, 39, 90, 95, 30

2, 3, 25, 30, 39, ~~25~~ 90, 95

Ans: 4 comparisons and 6 shifts

~~4 shift and~~




4 Comparison

3 Shift

0

3. For each of the code blocks below, write down the Big-Oh complexity analysis. (3 marks)

| | |
|---|---|
| **Case1:**<br>`for(i = 0; i < 50*(n+n+n); i++) {`<br>`  for(j = 0; j < i; j++){`<br>`  // do a simple swap operation`<br>`  }`<br>`}` | $O(N^2)$ |
| **Case2:**<br>`for(i = 0; i < pow(n,4); i++) {`<br>`  // do sequence of statements`<br>`  // each having constant time`<br>`}` | $O(N^4)$ |
| **Case 3:**<br>`int i = n;`<br>`while (i > 0) {`<br>`  //print statements`<br>`  i = i / 2;`<br>`}` | $O(\log n)$ |

4. Let A = [3a, 2, 1a, 5, 10, 3b, 6, 4, 8, 1b] be an array where 3a=3b and 1a = 1b and the letters a, b are used only to denote the order in which they occur in the input. Write down the values of A for each pass, when you sort it using selection sort. (5 marks)

A = [3a, 2, 1a, 5, 10, 3b, 6, 4, 8, 1b]

3a = 3b
1a = 1b

A = [2, 1a, 1b, 1c

1 → 1a, 2, 3a, 5, 10, 3b, 6, 4, 8, 1b

2 → 1a, 1b, 3a, 5, 10, 3b, 6, 4, 8, 2

3 → 1a, 1b, 2, 5, 10, 3b, 6, 4, 8, 3a

4 → 1a, 1b, 2, 3b, 10, 5, 6, 4, 8, 3a

5 → 1a, 1b, 2, 3b, 3a, 5, 6, 4, 8, 10

6 → 1a, 1b, 2, 3b, 3a, 4, 6, 5, 8, 10

7 → 1a, 1b, 2, 3b, 3a, 4, 5, 6, 8, 10

1a, 1b, 2, 3b, 3a, 4, 5, 6, 8, 10

5. Consider the following memory dump which you should be familiar with.

```
0x7fd7f0c02d20
  0  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31
F0 2B C0 F0 D7 7F 00 00 EE EE EE EE EE EE EE EE EE EE EE EE EE EE EE EE 00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00 AA AA AA AA AA AA AA AA AA AA AA AA AA AA AA AA DD DD EE EE EE EE EE EE
EE EE EE EE EE EE EE EE 00 00 00 00 00 00 00 00 DD DD 40 2D C0 F0 D7 7F 00 00 AA AA AA AA AA AA AA AA
AA AA AA AA AA AA AA AA DD DD EE EE EE EE EE EE EE EE EE EE EE EE 00 00 00 00 00 00 00 00 DD DD
70 2D C0 F0 D7 7F 00 00 AA AA AA AA AA AA AA AA AA AA AA AA AA AA AA AA DD DD EE EE EE EE EE EE
EE EE EE EE EE EE EE EE 00 00 00 00 00 00 00 00 DD DD A0 2D C0 F0 D7 7F 00 00 AA AA AA AA AA AA AA AA
AA AA AA AA AA AA AA AA DD DD EE EE EE EE EE EE EE EE EE EE EE EE EE EE EE EE 01 00 0F 00 00 00 01 DD DD
BB BB BB BB BB BB BB BB BB BB BB BB BB BB BB BB BB BB BB BB BB BB BB BB DD DD EE EE EE EE EE EE
EE EE EE EE EE EE EE EE 01 00 0E 00 00 00 01 DD DD BB BB BB BB BB BB BB BB BB BB BB BB BB BB BB BB
BB BB BB BB BB BB BB BB DD DD

0x7fd7f0c02bf0
  0  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31
00 00 00 00 00 00 00 00 EE EE EE EE EE EE EE EE EE EE EE EE EE EE EE EE 01 00 0D 00 00 00 01 DD DD
BB BB BB BB BB BB BB BB BB BB BB BB BB BB BB BB BB BB BB BB BB BB BB BB DD DD EE EE EE EE EE EE
EE EE EE EE EE EE EE EE 01 00 0C 00 00 00 01 DD DD BB BB BB BB BB BB BB BB BB BB BB BB BB BB BB BB
BB BB BB BB BB BB BB BB DD DD EE EE EE EE EE EE EE EE EE EE EE EE EE EE EE EE 01 00 0B 00 00 00 01 DD DD
BB BB BB BB BB BB BB BB BB BB BB BB BB BB BB BB BB BB BB BB BB BB BB BB DD DD EE EE EE EE EE EE
EE EE EE EE EE EE EE EE 01 00 0A 00 00 00 01 DD DD BB BB BB BB BB BB BB BB BB BB BB BB BB BB BB BB
BB BB BB BB BB BB BB BB DD DD EE EE EE EE EE EE EE EE EE EE EE EE EE EE EE EE 07 00 08 00 00 00 01 DD DD
BB BB BB BB BB BB BB BB BB BB BB BB BB BB BB BB BB BB BB BB BB BB BB BB DD DD EE EE EE EE EE EE
EE EE EE EE EE EE EE EE 02 00 09 00 00 00 01 DD DD BB BB BB BB BB BB BB BB BB BB BB BB BB BB BB BB
BB BB BB BB BB BB BB BB DD DD
```

Assume:

- we're on a 64-bit architecture
- the output above is from the DumpPages function in the driver
- debugging patterns in each byte are the same as in your assignment

```
static const unsigned char UNALLOCATED_PATTERN = 0xAA;
static const unsigned char ALLOCATED_PATTERN = 0xBB;
static const unsigned char FREED_PATTERN = 0xCC;
static const unsigned char PAD_PATTERN = 0xDD;
static const unsigned char ALIGN_PATTERN = 0xEE;
```

a. What is the value of PageSize_? (1 mark) 32 ✗ 298 (9 × 32) + 10 = 298

b. What is the value of ObjectsPerPage_? (1 mark) 4 ✗ 6

c. What is the value of ObjectsInUse_? (1 mark) 6 ✗ 8

d. What is the value of PadBytes_? (1 mark) 01 ✗ 2

e. What is the max usage count of a block? (1 mark) 4 ✗ 7

0