Date 3.2 No

```
void initialize (double **p) {
  for (int i=0 ; i<5 ; i++)
    *(p+i) = new double [i+1];
}
int main() {
  double *p[10];
  initialize (p+3);
  release(p);
}
```

a) Explain:

- Declare an array of 10 pointers point to double. No ~~dynamically~~ dynamic memory allocation in this line.

→ Memory allocated: 10 . size of (pointer) = 10 . 4 = 40 (Assume program is 32 bit)

- Initialize: (p+3) which means go to p[3]. For-loop will run from i=0 to 4, p[3] → p[7]

+ p[3]: 1 block memory: 1×4 = 4
+ p[4]: 2 blocks memory: 2×4 = 8
+ p[5]: 3 blocks memory: 3×4 = 12
+ p[6]: 4 blocks memory: 4×4 = 16
+ p[7]: 5 blocks memory: 5×4 = 20

Total bytes allocated from initialize function is:

4 + 8 + 12 + 16 + 20 = 60 byte.

⟹ Total bytes are allocated at each line of main is: 40 + 60 = 100 bytes

Case 2: pointer is 8 byte.

- Init an array of pointer to double: 10 × 8 = 80
- Initialize function:
  + p[3]: 1 × 8 = 8
  + p[4]: 2 × 8 = 16
  + p[5]: 3 × 8 = 24
  + p[6]: 4 × 8 = 32
  + p[7]: 5 × 8 = 40

  ⟹ Total: 128 bytes

⟹ Total bytes: 80 + 120 = 208 bytes.

b) 120 bytes total allocated ~~from~~ in Initialize function from heap if pointer has 4 byte, 200 bytes if 8 bytes.

c)
```
void dealocate(double **p){
  for (int i=0; i<5; i++){
    delete [] p[i+3];
    p[i+3] = nullptr;
  }
}
```

More for a)

4 bytes because create a copy pointer for initialize function and 4 bytes to create a traversal pointer ⟹ 8 bytes.

Case 1: 128 bytes (pointer has 4 bytes)

Case 2: 208 bytes (pointer has 8 bytes)