Part a) $O(\log(\log n))$

```
int i=2;
while(i<n)
{
  i=i*i;
}
```

→

```
for(int i=2; i<n; i=pow(i,2)) - Runs n times
{
  O(1);
}
```

$O(\log(\log(n)))$

Part b) $O(n^{7/2})$

```
0 for(int i=1; i<n; i++){                    - Runs n times
1   if((i%(int)sqrt(n)==0){                  - Runs √n times
2     for(int k=0; k<pow(i,3); k++){ - Runs n^3 time
3       O(1);
4 }}}
```

Because of 1 2 runs every $\sqrt{n}^{th}$ time.

$(n^3 \cdot n^{-1/2}) = (n^{6/2} \cdot n^{-1/2}) = n^{5/2}$

Runs n time because of 0

$(n \cdot n^{5/2}) = (n^{2/2} \cdot n^{5/2}) = n^{7/2}$

$O(n^{7/2})$

Part c) $O(n^2 \log n)$

```
for(int i=1; i<=n; i++){          - Runs n times
  for(int k=1; k<=n; k++){        - Runs n times
    if(A[k]==i){
      for(int m=1; m<=n; m=2m){} - Runs log(n) times
        O(1);
}}}}
```

$O(n^2 \log n)$

Part a) O(n)

```
int * a = new int[10];                                  - Runs 1 time
int size = 10;                                          - Runs 1 time
for(int i=0; i<n; i++){                                 - Runs 1 time
  if(i==size){                                          - Runs n times
    int newsize = 3*size/2;                             - Runs 1 times
    int* b = new int [newsize];                         - Runs 1 times
    for (int j=0; j<size; j++){ b[j]=a[j]}              - Runs size times
    delete []a                                          - Runs 1 times
    a=b;                                                - Runs 1 times
    size = newsize;                                     - Runs 1 times
    }                                                   - Runs 1 times
  a[i] = i*i;                                           - Runs 1 times
}}
```

O(n x size) ← n times constant

O(n)