Abdelrahman Adel Abdelfattah Sheet 6 No. 17012296 Date. / /

① a)
```
#Pragma Parallel{
  for (j=0; j<N; j++){
    Value t sum = 0
    for(K=0; K<L; K++)
        sum += A[get Thread x L + K] * B[K*N+j];
    C[get Thread x N+j] = sum;
  }
}
```

Parallelized the of each raw in A, since ther are no data dependancies in C[get_Thread * N + j]

b) 1- Can't parallize first loop S[] is dependant on each loop & the result of the previous loop

2- Parallelizing scond loop into one bigger loop

```
for (i=0; i<M; i++){
   # Porgma Parallel{
     if (get_thread < N-2){
       S[get_thread +2] = 0
       for(k=-2; k<3; k++)
          S[get thread +2] += 0.2* V[get thread + k];
     }
     v[get thread] = S[get Thread]
   }
```

No.
Date. / /

(2) a) Count the number of element that is less than the current element then put it in that element position

b) the outer most loop is the best one to parallelize

for(j=0;) ⇒ Depends on X[i]
for(i=0;) ⇒ Depends on nothing only temp array

There might be a race if the count is the same for two elements.