מבחן 2011 מועד ב׳:

1. א.1 בהנחה ש n≥m $n \cdot m$

״ ״ m>n n^2

ב. n

ג. $n \cdot m$ (לא בטוח)

ד. O(1)

<u>שאלה 2</u>:

א.

```
void flip_halves(int* v, int n)

    for(int i=0; i<n/2 ; i++){

        swap( a+i, a+n/2+i+n%2);

    }
    ~~if(n%2)~~
        ~~swap(a~~ [illegible]

}
void swap(int* a, int *b){

    int temp = *a;
    *a = *b;
    *b = temp;
}
```

Scanned by CamScanner

ב. נפתור את נוסחת הנסיגה:
פונקציה flip_halves יש לה סיבוכיות זמן O(n)
וסיבוכיות מקום O(1)

$$T(n) = \underbrace{n}_{\text{flip\_halves}} + \underbrace{2T(n/2)}_{\text{קוראים ל foo פעמיים עם אורך } n/2} =$$

$$= n + 2(n/2 + 2f(n/4)) = n + n + 4f(n/4)$$

$$= 2n + 4f(n/4) = 2n + 4\left(\frac{n}{4} + 2T\left(\frac{n}{8}\right)\right)$$

$$8n + 8T\left(\frac{n}{8}\right) = \ldots = n \cdot i + 2^i T\left(\frac{n}{2^i}\right)$$

ב i איטרציות;

עוצרים ב איטרציה: i = Log n

$$= n \operatorname{Log} n + 2^{\operatorname{Log} n} \cdot \underbrace{T\left(\frac{n}{2^{\operatorname{Log} n}}\right)}_{T(1)=1} = n \operatorname{Log} n + n =$$

$$= O(n \operatorname{Log} n)$$

סבי כית מקום: בגלל שאין הקצאות זכרון
אז סיבוכיות מקום היא עומק הרקורסיה
O(Log n)

שאלה 2

סעיף ג.

אחרי מעקב אחרי כמה דוגמאות אפשר לראות
שהפונקציה foo הופכת את המערך ולכן
נכתוב ~~את~~ ~~[illegible]~~ פונקציה שעושה את הכול בסיביוכיות
O(n) ומקום O(1)

```
void
~~int~~ foo2(int*v, intn){

        for(int i=0; i<n/2~~$~~;i++)

                swap(a+i ~~&~~a[n-1-i]);

}
```

כבר מימשנו את swap בסעיף קודם

Scanned by CamScanner

שאלה 3:

```
int return_occurences(int A[N][N], int x){

    int counter=0, i=0, j=0;

    while(A[0][j]<=x && j<N){

            if(A[0][j]==x){
                counter++; j--;
            }
            j++;
    }
         i++;
         if(j>0){ j--;}

    while(i<N){

         while(A[i][j]>=x){

              if(A[i][j]==x){
                  counter++;
              }
                if(j>0){j--;
                }else{ break;}
         }
         i++;

    }
    return counter;
}
```

4 : [illegible]

```
void print_all-solutions(int* dist, int* battery,
                                   int n){

        int energy = battery[0];
        int Sol [n] = {0} ;
        ~~int~~ Sol [0] = 1
      all-Sol-aux( dist, battery, n, energy, Sol, 0          //curr
                    100 );
                    //goal,
}
Void  all-Sol-aux(int* dist, int* battery, int n
                    int energy, int* ~~Sol~~, int goal))
                                     int curr
{     energy+= battery[curr];
    if( ~~goal~~ goal-energy <= 0){
           print_Stops( Sol, n);
           return;
      }
      if(curr== n)
          return;
          [illegible]
      for(int i=curr ; i<n; i++){

          if( energy < dist[i]- dist[curr])
                 continue;
           Sol[curr+1]=1;  energy -= dist[i]- dist[curr];
             all_sol_aux( dist , battery, n, energy
                              Sol, curr+1, ~~goal~~
                                             dist[curr+1]);
             sol[curr+1]=0; energy+=dist[i]- dist[curr];
        }

}
```