שאלה 5 (20 נקודות)

מעוניינים להרכיב צוות שמירה למוזיאון. ברחבי המוזיאון פזורות עמדות שמירה, שבכל אחת ניתן לשים שומר אחד בלבד. הדרישה היא שלכל פסל במוזיאון יהיו לפחות שני שומרים המסוגלים לראות אותו. ברצוננו למצוא את צוות השומרים **המינימאלי** הדרוש על מנת לשמור על המוזיאון. לשם כך, נתונה מטריצה a בגודל $M \times N$, כאשר N מייצג את מספר הפסלים ו-M את מספר עמדות השמירה. a[i][j] מכיל 1 אם נקודת השמירה ה- i משקיפה על הפסל ה- j, ו- 0 אחרת.

השלימו את הפונקציה findGuardTeam למטה, ואת פונקצית העזר בה היא משתמשת (בעמוד הבא), אשר מקבלת כפרמטר את המטריצה a ומחזירה את גודל צוות השמירה המינימלי המקיים את הדרישות. אם לא קיים כזה, הפונקציה מחזירה את הערך המיוחד M+1. כמו כן, במידה וניתן למצוא צוות שמירה העומד בדרישות, על הפונקציה לכתוב את מיקומי השומרים למערך result, שיכיל 1 במקום ה- i אם יש למקם שומר בעמדה ה- i, ו- 0 אחרת.

לנוחותכם, ניתן להשתמש בפונקצית העזר הבאה, המקבלת מערך באורך n של מספרים שלמים וערך x, ומחזירה 1 אם כל איברי המערך גדולים או שווים ל-x, ו-0 אחרת.

```
int atleastX(int arr[], int n, int x)
```

הערה: מספר השורות הריקות לא בהכרח מעיד על מספר שורות הקוד שיש לכתוב, ואורך הקו אינו בהכרח מעיד על אורך השורה שיש להשלים. ניתן להניח כי N ו-M הם קבועים המוגדרים כ-#define.

```
int findGuardTeam (int a[M][N], int result[M])
{
   int aux[_____N_____]= {0}, i;

   for ( i = 0 ; i < M ; ++i )
      result[i] = _______0_________;

   return guardTeamAux(a, result, aux, 0);
}
```

```
int guardTeamAux (int A[M][N], int result[M], int aux[], int k)
{
      int min_with_curr, min_without_curr, i;
      int res_with_curr[        M        ] = {0};

      if ( atleastX(aux, N, 2) == 1          ) return 0;
      if ( k == M                            ) return M+1;

      for ( i = 0 ; i < N ; ++i ) {
            aux[i] += A[k][i]                ;
      }

      min_with_curr =
            guardTeamAux ( A, res_with_curr, aux    , k+1   );

      for ( i = 0 ; i < N ; ++i ) {
            aux[i] -= A[k][i]                ;
      }
      min_without_curr =
            guardTeamAux ( A, result        , aux    , k+1   );

      if ( min_with_curr +1 < min_without_curr) {
            ____________________________________
            result[k] = 1;
            for (i=k+1; i<M; ++i) {
               result[i] = res_with_curr[i];
            }
            ____________________________________
            return min_with_curr + 1         ;
      }
      return min_without_curr          ;
}
```