```
a)
    vrstni red izpadanja za m=6, n=20
    5, 11, 17, 3, 10, 18, 6, 14, 2, 13, 4, 16, 9, 7, 1, 8, 15, 12, 0, 19

b) implementiral sem algoritem, ki gre skozi otroke toliko časa, dolker jih
    polovico ni izločenih.
    po tem se tisti otroci odstranijo (se naredi nov array in tastare prepišem)
    podobno se to zgodi tudi na tej polovici itd do števila npr 20 kjer je pa seznam
    že dovolj majhen

    časovna kompleksnost:
        seznam se vedno zmanjšuje na polovici -> zato logn * prepišemo logn elementov
        -> log^2 n
        izločiti pa moramo N otrok, pri katerih moramo iti log^2n krat čez while zanko
        ki povečuje temp_m
        rezultat -> n* log^2 n
    KODA V PRILOGI -> randm je nastavljen na FALSE zato se ne bo izvajal random M

c)
    časovna kompleksnot je podobna kot zgoraj, vendar M zakomplicira zadevo
    če bi bil M lahko še večji kot sem omejil tu (100), bi se notranja while
    zanka izvajala dlje časa -> N časa
    potem bi pa prišli do časovne kompleksnosti N**2

    KODA V PRILOGI -> randm je nastavljen na TRUE
```

```python
from random import randint
def incM(_m, n):  # increase m
    _m += 1
    if _m >= n:
        _m %= n
    return _m
def init():
    m, n = input("vpisite m in n\n").split()
    m, n = int(m), int(n)  # m je zmanjšan za 1 da dela lepo z indexi
    ##INIT ARRAYS
    ax_bool = []
    for i in range(n):  # init empty array with indices
        ax_bool.append(True)
    return m, n, ax_bool
def izstevanka(randm=False):
    koef = 2

    m, n, ax_bool = init()
    _n = n
    _m = m - 1
    while _n:  ##ODSTEVAJ
        ax_bool[_m] = False
        print(_m)  # taj biv izlocen

        if randm:  ##IZBERE NAKLJUČEN M.... LAHKO TUDI PREKO USER INPU AMPAK... TOO SLOW
            _m = randint(0, 100)  # predpostavka da otroci znajo steti do 100
            # _m = input("izpadli otrok, vpisi nov M") # <- z user inputom
            if _m >= n:
                _m %= n
        _n -= 1
        move_count = 0
        while (move_count < m):  # mormo premaknt vsaj m-krat
            _m = incM(_m, n)
            if ax_bool[_m] != False:  # izpadle otroke ignoriramo
                move_count += 1
            if _n == 0:
                return
        ################ IMPROVEMENT -> VSAKO POLOVICO ARRAYA GA PREPIŠEMO V NOVEGA##############
        if int(n / koef) > 20 and _n == int(n / koef): ##na polovici, potem četrtini, potem osmini še enkrat prepiše array
            ax_bool1 = []
            for i in range(int(n / koef)):  # init empty array with indices
                ax_bool1.append(True)
            n = len(ax_bool1)
            ax_bool = ax_bool1
            _m %= n  # še modulam da nebo out of bounds

            koef *=2 # da dobimo delitve še na četrtine itd
def main():
    izstevanka()
    izstevanka(True)
main()
```