EX.D

n, m, s = map(int, input().split())

min\_cuts = float('inf')

for x in range(min(n -1, 1000)): #Limit x to reduce computation time

for y in range(min(m - 1, 1000)): #Limit y to reduce computation time

total\_cells = (x + 1) \* (y + 1) \* n \* m // ((x+1) \* (y+1))

if total\_cells >= s:

min\_cuts = min(min\_cuts, x + y)

if min\_cuts == float('inf'):

print(-1) #No solution found within the limited search space

else:

print(min\_cuts)

Бұл жетілдірілген код x және y үшін іздеу кеңістігін шектеу арқылы уақыттың күрделілігі мәселесін шешеді. мәселенің күрделілігі кірістірілген ілмектерден туындайды; шектеусіз, үлкен n және m үшін өте ұзақ уақыт кетуі мүмкін. ілмектерді ақылға қонымды санмен шектеу арқылы (бұл жағдайда 1000, қажет болған жағдайда реттеңіз), біз есептеу уақытын айтарлықтай қысқартамыз, сонымен бірге егер ол осы ауқымда болса, шешім табудың жоғары мүмкіндігіне ие боламыз. Код әлі де тік және көлденең кесінділердің (x және y) ықтимал сандары арқылы қайталанады. Әрбір комбинация үшін ол кесілгеннен кейін қол жетімді ұяшықтардың жалпы санын есептейді. Егер жиынтық қажетті s-ден үлкен немесе оған тең болса, ол min\_cuts-ті осы уақытқа дейін табылған кесулердің ең аз санымен жаңартады. Егер іздеу кеңістігінде шешім табылмаса, ол -1 басып шығарады. Шектеу ретінде 1000-ды таңдау ерікті болып табылады; бұл мәнді енгізу шектеулеріне байланысты реттеу қажет болуы мүмкін. Өте үлкен кіріс үшін неғұрлым күрделі алгоритм қажет болуы мүмкін