محمد مهدی نظری – ۹۹۳۱۰۶۱

در فایل ارسال شده پروژه پایتون و فایل توضیحات بصورت پی دی اف و ورد و کد پایتون بصورت تکست (برای اینکه اگر کد مشکل برخورد) و همچنین اسکرین شات های مثال داده شده در فایل زیپ به ترتیب شماره گذاری شده اند .

توضیح الگوریتم :

برای اجرای الگوریتم حذف گاوسی باید ماتریس اولیه را درصورت امکان با عملیات سطری (جمع زدن یک سطر با ترکیب خطی سطور دیگر یا جابجایی دوسطر یا ضرب یک عدد در یک سطر) به شکل پلکانی کاهش یافته در بیاوریم یعنی درایه های قطر اصلی ۱ و بقیه صفر بشوند .برای اینکار از اولین درایه قطر اصلی شروع میکنیم و اگر ۰ نبود کل سطر را تقسیم بر آن درایه میکنیم تا یک شود .سپس برای اینکه درایه های زیر قطر اصلی صفر شود از سطر بعدی تا آخر هر بار حاصلضرب سطر مربوطه (که درایه قطر اصلی درآن بود) در درایه زیر قطر اصلی را از همان سطر کم میکنیم و این کاررا تا سطر و درایه آخر قطر اصلی تکرار میکنیم . به عنوان مثال در دستگاه داده شده اول ۲- برابر سطر اول را از سطر دوم بعد ۰ برابر سطر اول از سطر سوم را کم میکنیم و به همین ترتیب تا سطر آخر پیش میرویم و برای همه درایه های قطر اصلی این کاررا تکرار میکنیم .

حال اگر درایه قطر اصلی ۰ بود باید سعی کنیم آن سطر را با سطر بعدی که درایه زیر قطر اصلی ۰ نیست عوض کنیم و سپس کارهای بخش قبل را انجام دهیم . اگر عنصر غیر ۰ هم موجود نبود کاری با آن سطر نمی کنیم .مجموعه این کار ها فاز اول یا روبه جلو نام دارد که در کد زیر قابل مشاهده است .

از متغیر temp برای رفتن به خط های بعدی و از counter برای چک کردن اینکه همه ی سطر های بعدی عناصرمربوطه صفر باشد استفاده میکنیم .

```
i = 0
while i < n:
    if augmented_matrix[i][i] != 0:
        factor = augmented_matrix[i][i]
        for k in range(i, n + 1):
    # 1   kardane zarib moteghayer ha dar har khat
            augmented_matrix[i][k] = augmented_matrix[i][k] / factor
        print(augmented_matrix)
```

```
        print()
        for j in range(i + 1, n):
            factor2 = augmented_matrix[j][i]
            for r in range(i, n + 1):
           # 0 kardane zir zarib motaghaier
                augmented_matrix[j][r] -= augmented_matrix[i][r] * factor2
            print(augmented_matrix)
            print()
        i += 1
    else:
        if i < n - 1:
            temp = i + 1
            counter = 0
            while augmented_matrix[temp][i] == 0 and counter != n - 1 - i:
                counter += 1
                if temp < n - 1:
                    temp += 1
            if counter != n - 1 - i:
#                   swap row
                augmented_matrix[[i, temp]] = augmented_matrix[[temp, i]]
                print(augmented_matrix)
                print()
            else:
                i += 1
                print(augmented_matrix)
                print()
        else:
```

```
        i += 1
        print(augmented_matrix)
        print()
```

در فاز دوم یا برگشتی هم باید درایه های بالای قطر اصلی را صفر کنیم که کار نسبت به بخش قبل راحتتر است .برای اینکار از سطر و ستون یکی مانده به آخر در ماتریس افزوده شروع میکنیم و بالا میایم و درایه های مربوطه را صفر میکنیم و همزمان در هر سطر حاصلضرب درایه مربوطه در درایه ستون آخر سطر مرتبط با قطر اصلی را از درایه ستون آخر همان سطر کم میکنیم (مثلا فرض کنید در حلقه تودرتو در درایه با سطر ۳ و ستون ۴ ماتریس افزوده هستیم که ماتریس ۵ در ۶ است (شماره سطر و ستون از ۰ شروع میشوند) . حال اول مقدار آن درایه را در ستون آخر سطر ۴ ضرب میکنیم و بعد از درایه ستون آخر سطر ۳ کم میکنیم در نهایت هم مقدار درایه اولیه را ۰ میکنیم) . البته اگر درایه روی قطر اصلی در یک ستون ۱ نباشد یعنی ۰ باشد آنگاه درایه های بالای آن را نمی توان صفر کرد و همان مقداری که در فاز اول داشتند را حفظ میکنند .

```
for ii in range(n - 2, -1, -1):
    m = n - 1
    for jj in range(n - 1, ii, -1):
        fac = augmented_matrix[ii][jj]
        augmented_matrix[ii][n] -= fac * augmented_matrix[jj][n]
#         agar deraye ghotr asli marboote 1 nabood balayee ra 0 nakonad
        if augmented_matrix[m][m] == 1:
            augmented_matrix[ii][jj] = 0
        m -= 1
    print(augmented_matrix)
    print()
```

حال که ماتریس افزوده ساخته شده سه حالت برای جواب نهایی داریم .

۱ . اگر سطری باشد که همه ی درایه ها غیر از ستون آخر صفر باشد قطعا جواب نداریم .

۲ – اگر حالت قبل رخ ندهد و سطر کلا صفر داشته باشیم دستگاه بینهایت جواب دارد چونکه ماتریس اولیه قطعا مربعی است و اگر سطر صفر داشته باشیم حداقل یک متغیر آزاد داریم .

۳ – دو حالت قبل رخ ندهد در این صورت یک جواب داریم که همان ستون آخر است .

حالات زیر در کد چک شده و پیغام مناسب برای کاربر چاپ میکند . همچنین در این بخش از تابعی استفاده شده که تعداد صفر های یک سطر ماتریس را میشمارد :

```
def count_zeros(number):
    #  return str(number).count('0.')
    t = 0
    for u in number:
        if u == float(0):
            t += 1
    return t
inconsistent = 0
consistent = 0
for rr in range(n):
    if count_zeros(augmented_matrix[rr]) == n and augmented_matrix[rr][n] != 0:
        inconsistent = 1
    if count_zeros(augmented_matrix[rr]) == n + 1:
        consistent = 1
if inconsistent == 1:
    print(" dastgah javab nadarad ")


if inconsistent == 0 and consistent == 1:
    print(" dastgah binahayat javab darad ")


if inconsistent == 0 and consistent == 0:
    print(" dastgah tak javab be shekl zir darad ")
    print(augmented_matrix[:, n])
```