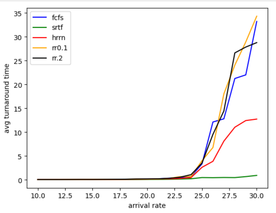
In [28]:
```
import pandas as pd
import matplotlib.pyplot as plt
%matplotlib inline


fcfs = pd.read_csv('Sim/simdata_fcfs.csv')
srtf = pd.read_csv('Sim/simdata_srtf.csv')
hrrn = pd.read_csv('Sim/simdata_hrrn.csv')
rr1 = pd.read_csv('Sim/simdata_rr_01.csv')
rr2 = pd.read_csv('Sim/simdata_rr_02.csv')
```

In [29]:
```
print(fcfs)
```

```
    arrival_rate  service rate  avg turnaround  total throughput  cpu util  \
0             10          25.0        0.066880         10.234223  0.404404
1             11          25.0        0.071106         10.909011  0.434434
2             12          25.0        0.074153         11.762535  0.465401
3             13          25.0        0.080276         12.900523  0.516144
4             14          25.0        0.089149         13.830746  0.554534
5             15          25.0        0.099613         14.791124  0.588801
6             16          25.0        0.109013         15.976153  0.637995
7             17          25.0        0.115242         16.958142  0.674006
8             18          25.0        0.138772         17.962208  0.720504
9             19          25.0        0.151015         19.053566  0.747921
10            20          25.0        0.180845         19.712473  0.784614
11            21          25.0        0.224545         20.847494  0.837035
12            22          25.0        0.375193         21.957245  0.865872
13            23          25.0        0.490250         23.211883  0.928797
14            24          25.0        0.693975         23.699073  0.944296
15            25          25.0        3.351606         24.431284  0.988420
16            26          25.0       12.143129         25.057347  0.999355
17            27          25.0       12.808251         25.303036  0.997327
18            28          25.0       21.261573         24.960237  0.999945
19            29          25.0       22.022796         25.418182  0.999813
20            30          25.0       33.225825         25.194383  0.999811

    avgprocesses
0       1.292400
1       1.345031
2       1.465956
3       1.585349
4       1.666200
5       1.880512
6       2.184179
7       2.253275
8       2.742526
9       3.122500
10      3.780988
11      4.786321
12      8.551600
13     31.361255
14     16.626306
15     83.607668
16    321.113669
17    346.894946
18    591.896172
19    634.958393
20   1001.412828
```

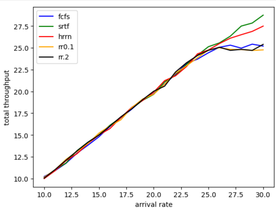
In [30]:
```
x = fcfs['arrival_rate']
avg_turnaround_fcfs = fcfs['avg turnaround']
total_throughput_fcfs = fcfs['total throughput']
cpu_util_fcfs = fcfs['cpu util']
avg_processes_fcfs = fcfs['avgprocesses']

avg_turnaround_srtf = srtf['avg turnaround']
total_throughput_srtf = srtf['total throughput']
cpu_util_srtf = srtf['cpu util']
avg_processes_srtf = srtf['avgprocesses']

avg_turnaround_hrrn = hrrn['avg turnaround']
total_throughput_hrrn = hrrn['total throughput']
cpu_util_hrrn = hrrn['cpu util']
avg_processes_hrrn = hrrn['avgprocesses']

avg_turnaround_rr1 = rr1['avg turnaround']
total_throughput_rr1 = rr1['total throughput']
cpu_util_rr1 = rr1['cpu util']
avg_processes_rr1 = rr1['avgprocesses']

avg_turnaround_rr2 = rr2['avg turnaround']
total_throughput_rr2 = rr2['total throughput']
cpu_util_rr2 = rr2['cpu util']
avg_processes_rr2 = rr2['avgprocesses']

plt.plot(x, avg_turnaround_fcfs,color='b')
plt.plot(x, avg_turnaround_srtf,color='g')
plt.plot(x, avg_turnaround_hrrn,color='r')
plt.plot(x, avg_turnaround_rr1,color='orange')
plt.plot(x, avg_turnaround_rr2,color='black')
location = 0 # For the best location
legend_drawn_flag = True
plt.legend(["fcfs", "srtf", "hrrn", "rr0.1", "rr.2"], loc=0, frameon=legend_drawn_flag)
plt.xlabel('arrival rate')
plt.ylabel('avg turnaround time')
plt.show()
```



AVG TURNAROUND TIME

As would be expected, SRTF has the lowest turnaround time. It's surprising, however, how much better it is than the other algorithms. HRRN also performs significantly better than the other 3.

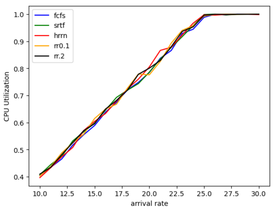
In [31]:
```
plt.plot(x, total_throughput_fcfs,color='b')
plt.plot(x, total_throughput_srtf,color='g')
plt.plot(x, total_throughput_hrrn,color='r')
plt.plot(x, total_throughput_rr1,color='orange')
plt.plot(x, total_throughput_rr2,color='black')
location = 0 # For the best location
legend_drawn_flag = True
plt.legend(["fcfs", "srtf", "hrrn", "rr0.1", "rr.2"], loc=0, frameon=legend_drawn_flag)
plt.xlabel('arrival rate')
plt.ylabel('total throughput')
plt.show()
```



TOTAL THROUGHPUT

SRTF and HRRN have the highest total throughput. This makes sense, they have the lowest average turnaround time. However the algorithms do not differ much in total throughput, only diverging at higher loads.

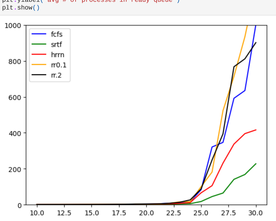
In [32]:
```
plt.plot(x, cpu_util_fcfs,color='b')
plt.plot(x, cpu_util_srtf,color='g')
plt.plot(x, cpu_util_hrrn,color='r')
plt.plot(x, cpu_util_rr1,color='orange')
plt.plot(x, cpu_util_rr2,color='black')
location = 0 # For the best location
legend_drawn_flag = True
plt.legend(["fcfs", "srtf", "hrrn", "rr0.1", "rr.2"], loc=0, frameon=legend_drawn_flag)
plt.xlabel('arrival rate')
plt.ylabel('CPU Utilization')
plt.show()
```



CPU UTILIZATION

All the algorithms have similar CPU utilization. Scales linearly up to nearly 100% at high loads

In [21]:
```
plt.ylim(top=1000)
plt.plot(x, avg_processes_fcfs,color='b')
plt.plot(x, avg_processes_srtf,color='g')
plt.plot(x, avg_processes_hrrn,color='r')
plt.plot(x, avg_processes_rr1,color='orange')
plt.plot(x, avg_processes_rr2,color='black')
location = 0 # For the best location
legend_drawn_flag = True
plt.legend(["fcfs", "srtf", "hrrn", "rr0.1", "rr.2"], loc=0, frameon=legend_drawn_flag)
plt.xlabel('arrival rate')
plt.ylabel('avg # of processes in ready queue')
plt.show()
```

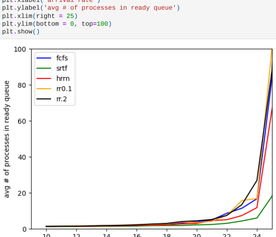


AVG NUMBER OF PROCESSES IN READY QUEUE

SRTF does the best job at keeping the ready queue empty, followed by HRRN. The other algorithms experience skyrocketing ready queue lengths at higher loads.

Since the queue length grows quickly, lets look at the behavior at lower loads more closely.

In [43]:
```
plt.plot(x, avg_processes_fcfs,color='b')
plt.plot(x, avg_processes_srtf,color='g')
plt.plot(x, avg_processes_hrrn,color='r')
plt.plot(x, avg_processes_rr1,color='orange')
plt.plot(x, avg_processes_rr2,color='black')
location = 0 # For the best location
legend_drawn_flag = True
plt.legend(["fcfs", "srtf", "hrrn", "rr0.1", "rr.2"], loc=0, frameon=legend_drawn_flag)
plt.xlabel('arrival rate')
plt.ylabel('avg # of processes in ready queue')
plt.xlim(right = 25)
plt.ylim(bottom = 0, top=100)
plt.show()
```



INTERPRETATION OF RESULTS

SRTF performs the best on all our metrics, followed by HRRN. FCFS and round robin could offer other advantages that aren't measured by this simulation.

In [ ]: