In [1]:
```python
import pandas as pd
import seaborn as sea
import matplotlib.pyplot as mtn
import numpy as np
import psycopg2
```

In [2]:
```python
db = psycopg2.connect(
    host='localhost',
    user='postgres',
    password='SQUAL#QUE',
    database='Banglore'
)

cur = db.cursor()
```

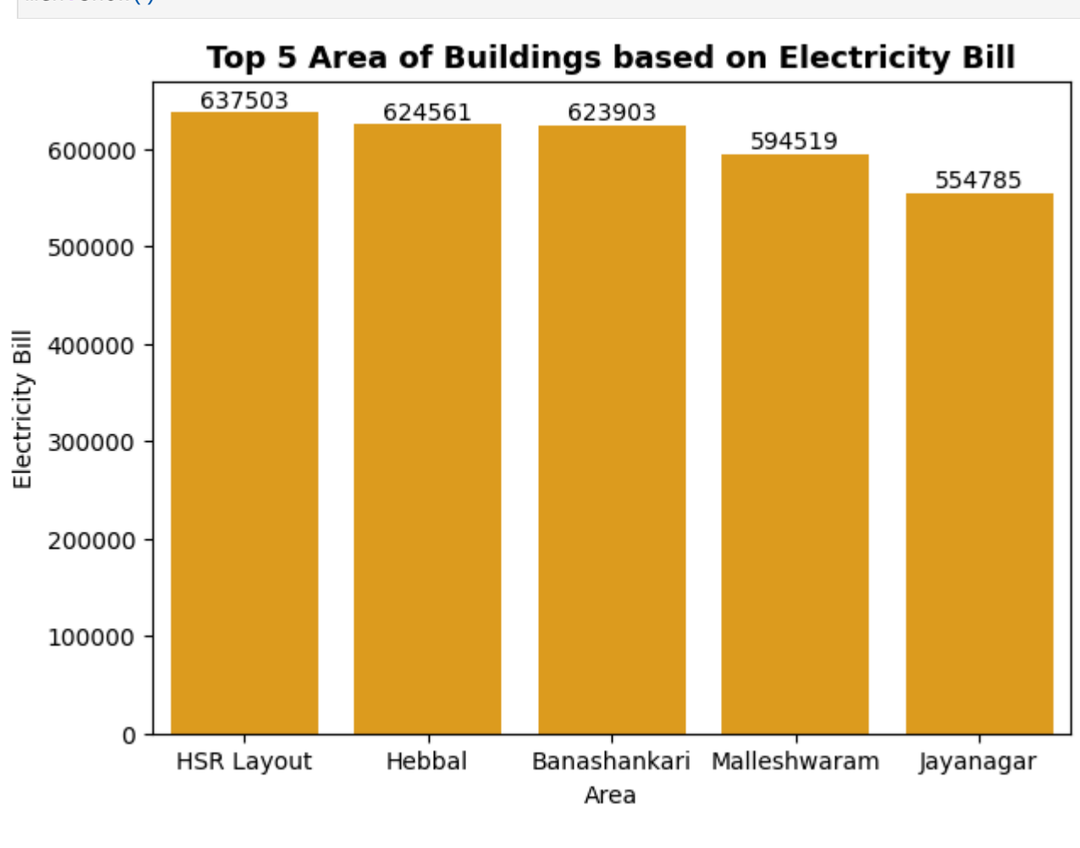
1. Retrieve the top 5 buildings with the highest electricity bill, including their city, area, and building type.

In [8]:
```python
query = """ select city, area, building_type, round(sum(cast (electricity_bill as decimal)),1) as Total_bill
from buildings
group by 1,2,3
order by 4 desc
limit 5 """
cur.execute(query)
data = cur.fetchall()
df = pd.DataFrame(data, columns = (['City', 'Area', 'Building_type', 'Electricity Bill']))
df
```

Out[8]:

	City	Area	Building_type	Electricity Bill
0	Bangalore	HSR Layout	Industrial	637503.4
1	Bangalore	Hebbal	Industrial	624560.7
2	Bangalore	Banashankari	Institutional	623903.2
3	Bangalore	Malleshwaram	Institutional	594519.3
4	Bangalore	Jayanagar	Residential	554785.4

In [15]:
```python
mtn.figure(figsize = (7,5))
ax = sea.barplot(x = 'Area', y = 'Electricity Bill', data = df, color = 'orange')
ax.bar_label(ax.containers[0])
mtn.title('Top 5 Area of Buildings based on Electricity Bill', fontsize = 13, fontweight = 'bold')
mtn.show()
```



2. Calculate the average energy consumption per square meter for each building type, ordered by the highest average.

In [20]:
```python
query = """ select building_type, round(avg(cast(energy_consumption_per_sqm as decimal)),2) as avg_energy_consumption
from buildings
group by 1
order by 2 desc"""
cur.execute(query)
data = cur.fetchall()
df = pd.DataFrame(data, columns = (['Industry Type', 'Avg_Energy_Consumption']))
df
```

Out[20]:

	Industry Type	Avg_Energy_Consumption
0	Residential	111.06
1	Commercial	109.36
2	Industrial	108.53
3	Institutional	106.15

3. Find the buildings where the occupancy rate is below the average occupancy rate of all buildings.

In [22]:
```python
query = """ select area, round(cast(occupancy_rate as decimal),2) as occupancy_rate
from buildings
where occupancy_rate > (select avg(occupancy_rate) from buildings)"""
cur.execute(query)
data = cur.fetchall()
df = pd.DataFrame(data, columns = (['Building areas', 'Occupancy Rate']))
df
```

Out[22]:

	Building areas	Occupancy Rate
0	Yelahanka	81.81
1	HSR Layout	96.51
2	Indiranagar	95.06
3	Whitefield	88.81
4	Hebbal	99.28
...	...	...
708	Yelahanka	84.19
709	Malleshwaram	84.91
710	Jayanagar	97.51
711	Koramangala	94.05
712	Yelahanka	77.50

713 rows × 2 columns

4. Identify the buildings that were last inspected before 2023 and have a maintenance priority of 'High'.

In [24]:
```python
query = """ select area, last_inspection_date, maintenance_priority
from buildings
where extract(year from last_inspection_date) < 2022 AND
maintenance_priority = 'High'"""
cur.execute(query)
data = cur.fetchall()
df = pd.DataFrame(data, columns = (['Area', 'Last_inspection_date', 'maintenance_priority']))
df.head(15)
```

Out[24]:

	Area	Last_inspection_date	maintenance_priority
0	Jayanagar	2021-12-09	High
1	Koramangala	2021-09-19	High
2	Malleshwaram	2021-08-11	High
3	Yelahanka	2021-11-14	High
4	BTM Layout	2021-12-30	High
5	HSR Layout	2021-10-16	High
6	Electronic City	2021-10-14	High
7	Whitefield	2021-09-20	High
8	Hebbal	2021-11-21	High
9	HSR Layout	2021-11-13	High
10	Indiranagar	2021-09-05	High
11	Whitefield	2021-11-30	High
12	Malleshwaram	2021-09-14	High
13	Banashankari	2021-12-16	High
14	BTM Layout	2021-09-26	High

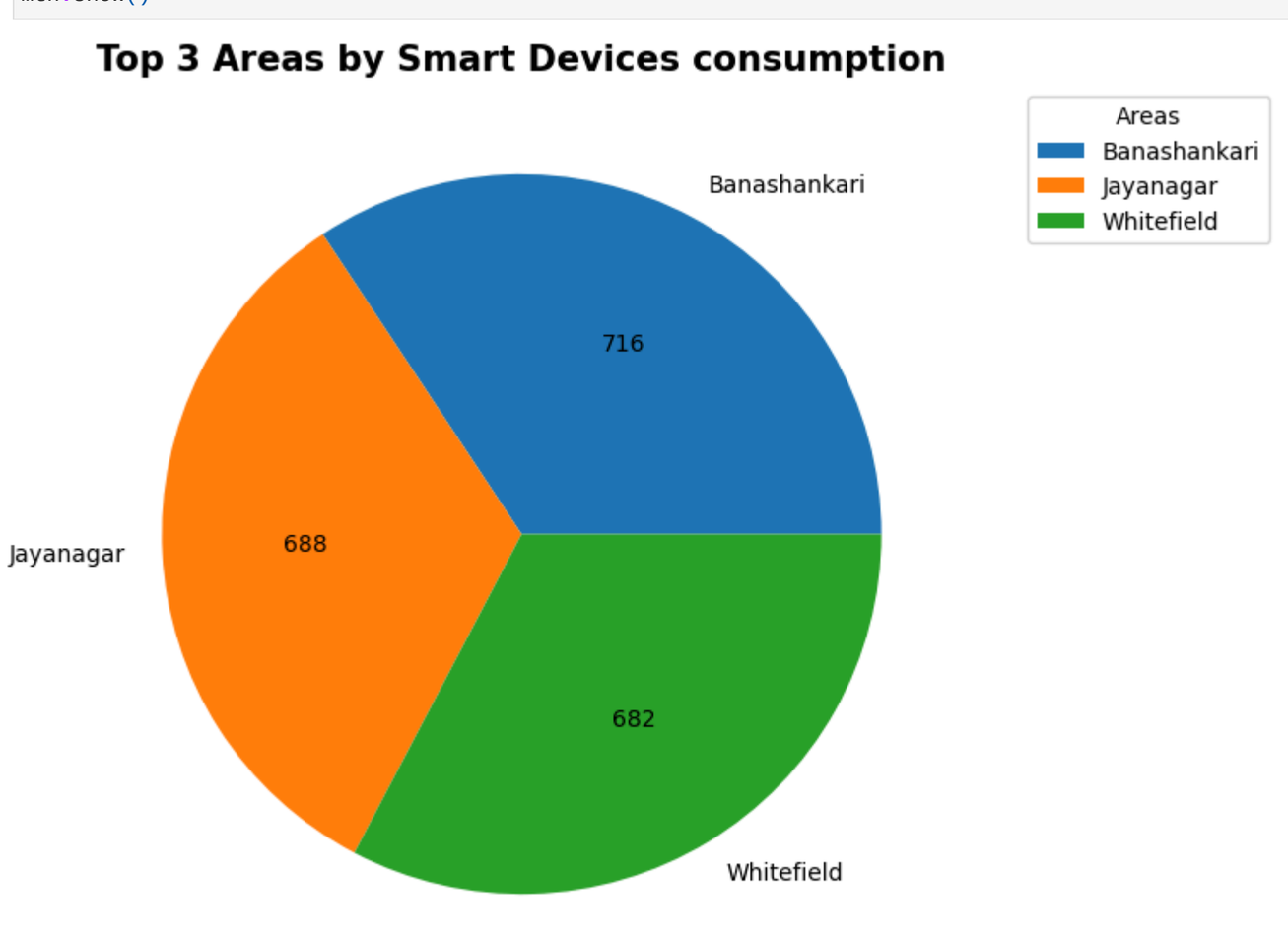
5. Retrieve the top 3 areas where the total number of smart devices across all buildings is the highest.

In [25]:
```python
query = """ select area, sum(smart_devices_count) as total_smart_devices
from buildings
group by 1
order by 2 desc
limit 3"""
cur.execute(query)
data = cur.fetchall()
df = pd.DataFrame(data, columns = (['area', 'Total_smart_devices']))
df
```

Out[25]:

	area	Total_smart_devices
0	Banashankari	716
1	Jayanagar	688
2	Whitefield	682

In [33]:
```python
mtn.figure(figsize = (8,7))
mtn.pie(df['Total_smart_devices'], autopct=lambda p: '{:.0f}'.format(p * sum(df['Total_smart_devices']) / 100), labels = df['area'])
mtn.title('Top 3 Areas by Smart Devices consumption', fontsize = 15, fontweight = 'bold')
mtn.legend(df['area'], title = 'Areas', bbox_to_anchor = (1.05,1), loc = 'upper left')
mtn.show()
```



6. Calculate the total water usage per building type and list the building types with total water usage greater than 50000.

In [35]:
```python
query = """ select * from
(select area, sum(water_usage_per_building) as total_water_usage
from buildings
group by 1)as d
where total_water_usage > 50000"""
cur.execute(query)
data = cur.fetchall()
df = pd.DataFrame(data, columns = (['area', 'Total_Water_Usage']))
df
```

Out[35]:

	area	Total_Water_Usage
0	Malleshwaram	70562.043332
1	Electronic City	63404.411396
2	Indiranagar	67779.629954
3	Hebbal	69958.987354
4	Koramangala	62123.161396
5	Banashankari	82830.572172
6	Whitefield	81870.561197
7	Yelahanka	78609.420381
8	HSR Layout	74509.067533
9	BTM Layout	71546.454617
10	Jayanagar	85582.136229

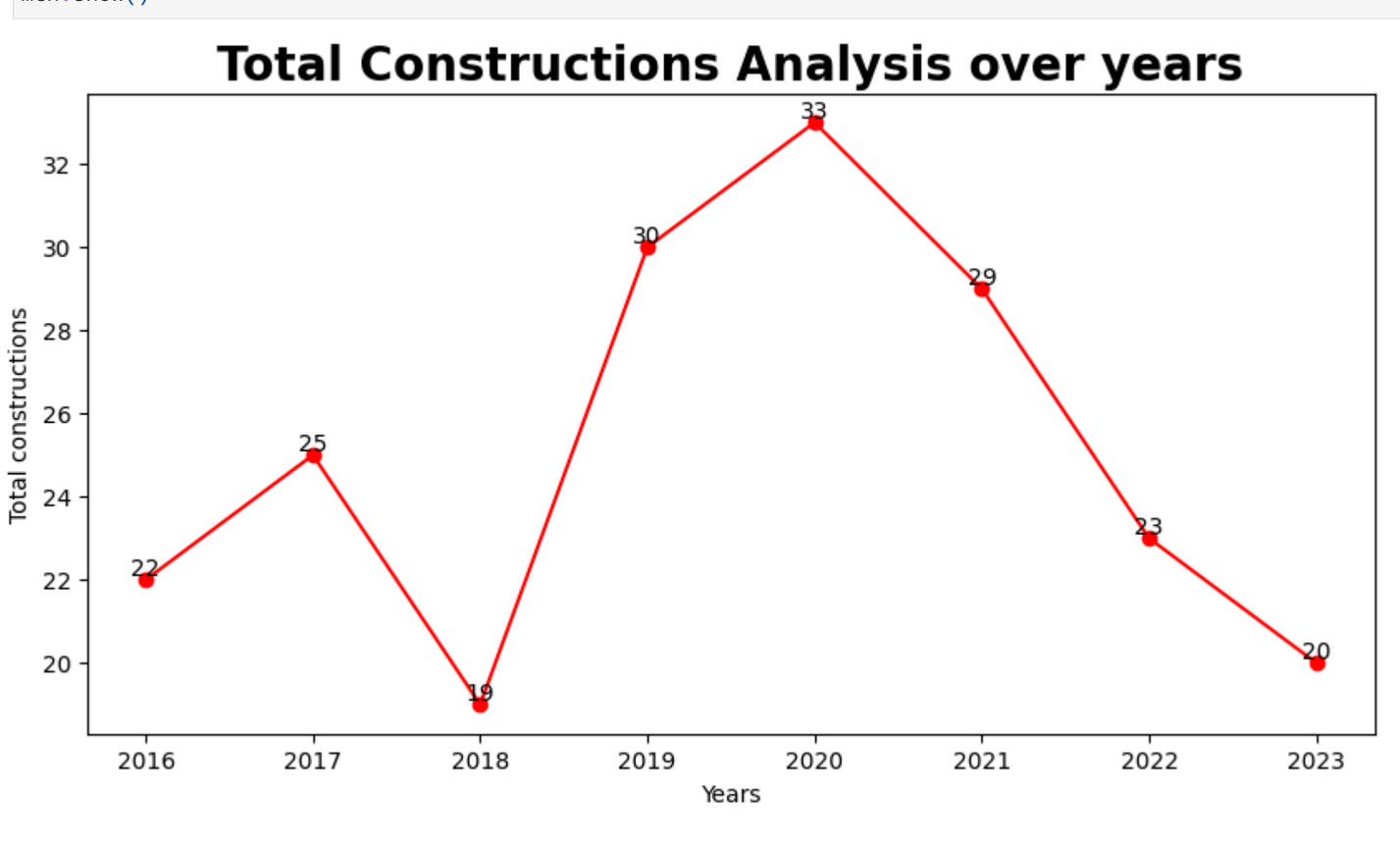
7. Analyse the Total Constructions held from 2015 to 2023

In [42]:
```python
query = """ select construction_year, count(*) as total_constructions
from buildings
where construction_year > 2015
group by 1
order by 1"""
cur.execute(query)
data = cur.fetchall()
df = pd.DataFrame(data, columns = (['Years', 'Total_constructions_held']))
df
```

Out[42]:

	Years	Total_constructions_held
0	2016	22
1	2017	25
2	2018	19
3	2019	30
4	2020	33
5	2021	29
6	2022	23
7	2023	20

In [52]:
```python
mtn.figure(figsize = (10,5))
mtn.plot(df['Years'], df['Total_constructions_held'], marker = 'o', color = 'r')
for x, y in zip(df['Years'], df['Total_constructions_held']):
    mtn.text(x, y, f'{y}', ha='center', va='bottom')
mtn.title('Total Constructions Analysis over years', fontsize = 20, fontweight = 'bold')
mtn.xlabel('Years')
mtn.ylabel('Total constructions')
mtn.show()
```



8. Identify buildings where the maintenance resolution time is higher than the average maintenance resolution time for all buildings in the same city.

In [7]:
```python
query = """ select area,
round(cast(maintenance_resolution_time as decimal),2) as maintenance_resolution_time
from buildings
where maintenance_resolution_time > (select avg(maintenance_resolution_time) from buildings)
"""
avg = """ select avg(maintenance_resolution_time) from buildings """
cur.execute(avg)
avgg = cur.fetchall()
cur.execute(query)
data = cur.fetchall()
df = pd.DataFrame(data, columns = (['area', 'maintenance_resolution_time']))
df
```

Out[7]:

	area	maintenance_resolution_time
0	Whitefield	34.14
1	Hebbal	11.54
2	Malleshwaram	11.11
3	HSR Layout	14.78
4	HSR Layout	15.20
...	...	...
385	Malleshwaram	22.52
386	HSR Layout	11.87
387	Jayanagar	12.90
388	Koramangala	12.54
389	Hebbal	14.15

390 rows × 2 columns

In [8]:
```python
print('The average Maintenance_resolution_time is', avgg)
```

```
The average Maintenance_resolution_time is [(9.617932608220002,)]
```

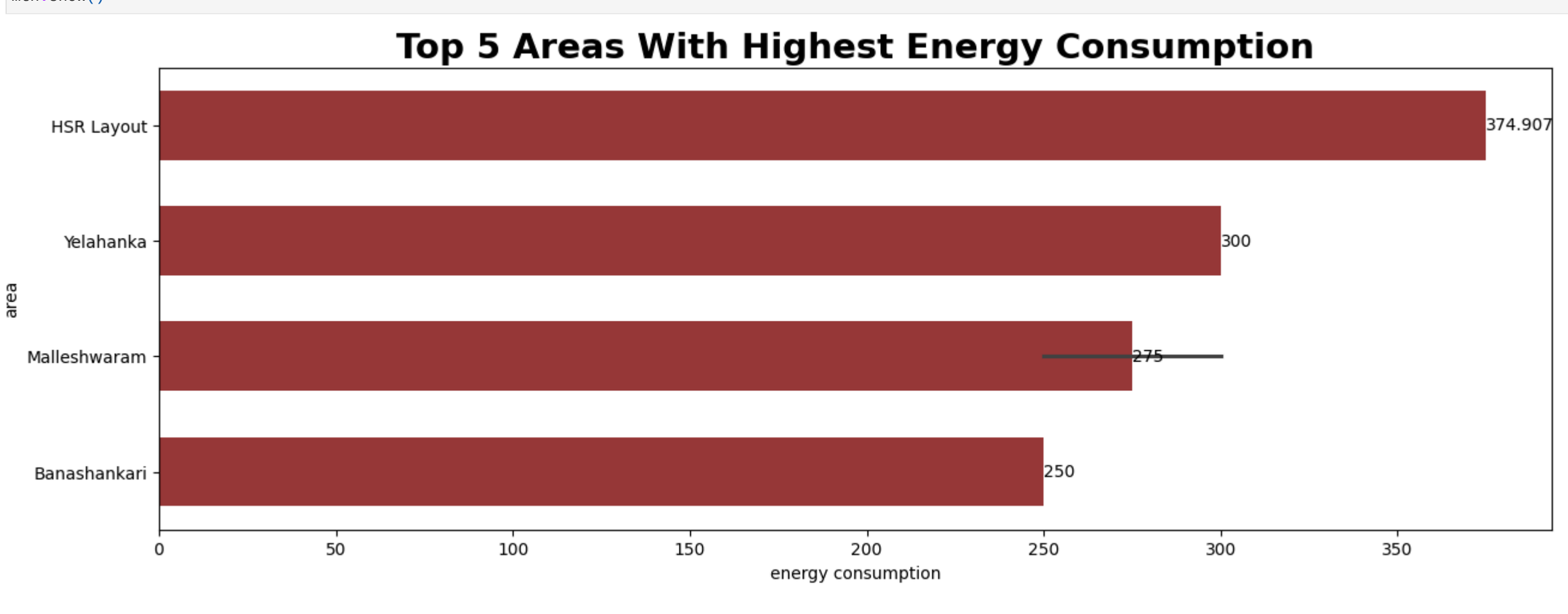
9. List the top 5 buildings with the highest energy consumption per square meter but with a waste recycling percentage below 30%

In [9]:
```python
query = """ select area, round(cast(waste_recycled_percentage as decimal),2) as Waste_recycled_percentage,
sum(energy_consumption_per_sqm) as total_energy_consumption
from buildings
where waste_recycled_percentage < 30
group by 1,2
order by 3 desc
limit 5"""
cur.execute(query)
data = cur.fetchall()
df = pd.DataFrame(data, columns = (['area', 'waste recycled', 'energy consumption']))
df
```

Out[9]:

	area	waste recycled	energy consumption
0	HSR Layout	15.04	374.90732
1	Yelahanka	11.11	300.00000
2	Malleshwaram	25.15	300.00000
3	Malleshwaram	13.47	250.00000
4	Banashankari	10.15	250.00000

In [23]:
```python
mtn.figure(figsize = (15,5))
ax = sea.barplot(y = 'area', x = 'energy consumption', data = df, color = 'brown', width = 0.6)
ax.bar_label(ax.containers[0])
mtn.title('Top 5 Areas With Highest Energy Consumption', fontweight = 'bold', fontsize = 21)
mtn.show()
```



10. Calculate the total number of residents for each area and building type, and display areas with a total resident count of more than 5000.

In [28]:
```python
query = """
select * from
(select area, building_type, sum(number_of_residents) as Total_residents
from buildings
group by 1,2) as q
where Total_residents > 7000
"""
cur.execute(query)
data = cur.fetchall()
df = pd.DataFrame(data, columns = (['area', 'buliding_type', 'total_residents']))
df.head(10)
```

Out[28]:

	area	buliding_type	total_residents
0	Whitefield	Residential	7382
1	Jayanagar	Commercial	7245
2	Koramangala	Institutional	7966
3	Yelahanka	Industrial	9205
4	Yelahanka	Commercial	7667
5	HSR Layout	Commercial	8176
6	Malleshwaram	Institutional	9859
7	BTM Layout	Institutional	7723
8	BTM Layout	Commercial	9198
9	Yelahanka	Residential	8468

11. Identify buildings constructed after the year 2000 that have an indoor air quality rating below the overall average.

In [3]:
```python
query = """ select area, construction_year, round(cast(indoor_air_quality as decimal),2) as indoor_air_quality
from buildings
where indoor_air_quality > (select avg(indoor_air_quality) from buildings) AND
construction_year > 2000 """
cur.execute(query)
data = cur.fetchall()
df = pd.DataFrame(data, columns = (['area', 'construction_year', 'indoor_air_quality']))
df
```

Out[3]:

	area	construction_year	indoor_air_quality
0	Yelahanka	2004	49.60
1	Malleshwaram	2017	79.92
2	Koramangala	2020	48.55
3	Hebbal	2017	57.74
4	Whitefield	2020	67.60
...	...	...	...
220	HSR Layout	2022	100.00
221	Banashankari	2017	77.34
222	HSR Layout	2018	49.85
223	Hebbal	2006	48.97
224	Jayanagar	2007	100.00

225 rows × 3 columns

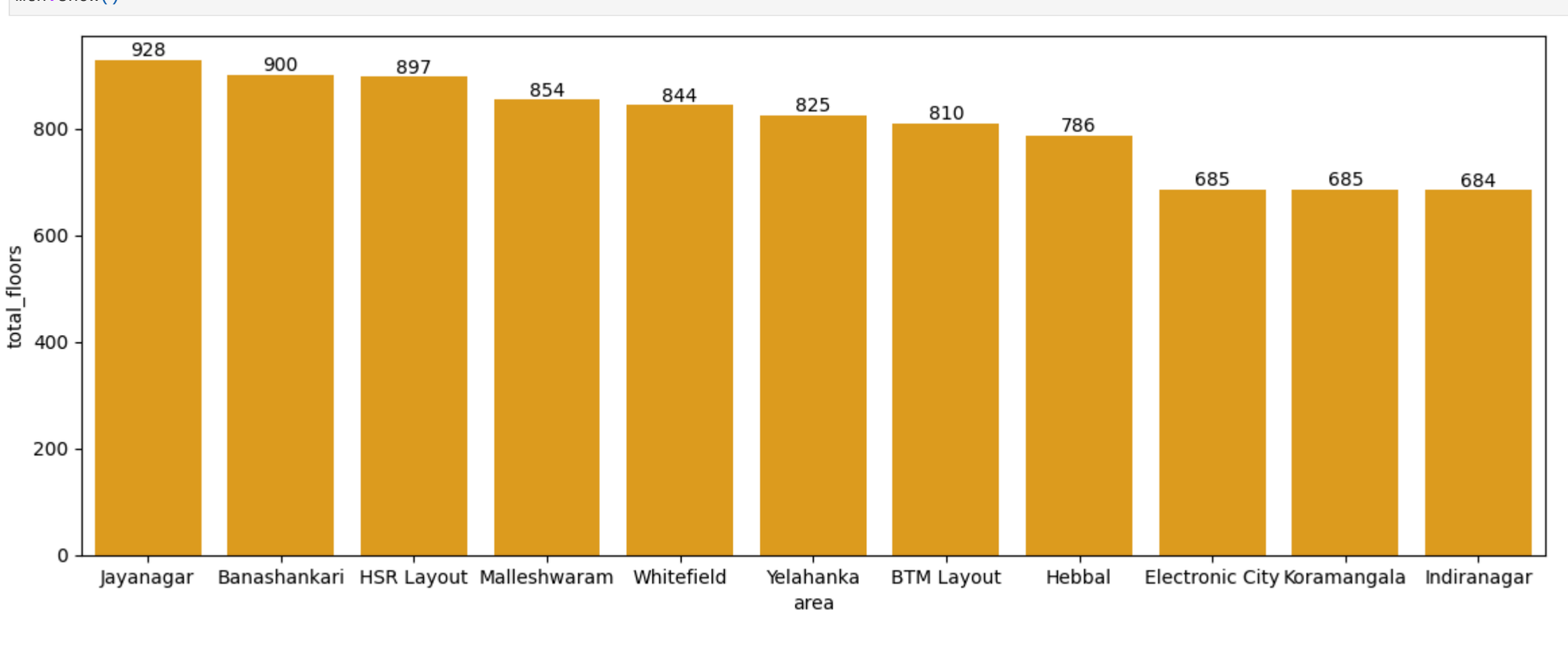
12. Find the building with the highest number of floors and the lowest occupancy rate.

In [4]:
```python
query = """ select area, sum(number_of_floors) as total_floors,
round(avg(cast(occupancy_rate as decimal)),2) as avg_occupancy_rate
from buildings
group by 1
order by 2 desc, 3 asc"""
cur.execute(query)
data = cur.fetchall()
df = pd.DataFrame(data, columns = (['area', 'total_floors', 'avg_occupancy_rate']))
df
```

Out[4]:

	area	total_floors	avg_occupancy_rate
0	Jayanagar	928	76.85
1	Banashankari	900	75.95
2	HSR Layout	897	78.58
3	Malleshwaram	854	77.51
4	Whitefield	844	76.94
5	Yelahanka	825	78.88
6	BTM Layout	810	75.39
7	Hebbal	786	78.74
8	Electronic City	685	74.78
9	Koramangala	685	77.76
10	Indiranagar	684	78.52

In [6]:
```python
mtn.figure(figsize = (14,5))
ax = sea.barplot(x = 'area', y = 'total_floors', data = df, color = 'orange')
ax.bar_label(ax.containers[0])
mtn.show()
```



13. Calculate the correlation between energy consumption per square meter and the number of smart devices in each building.

In [13]:
```python
query = """ select corr(energy_consumption_per_sqm, number_of_floors) as correlations
from buildings"""
cur.execute(query)
data = cur.fetchall()
df = pd.DataFrame(data, columns = (['Correlation']))
df
```

Out[13]:

	Correlation
0	0.034977

correlation value between energy_consumption_per_sqm and number of floors is 0.0349, this indicates an almost negligible correlation between the two variables.

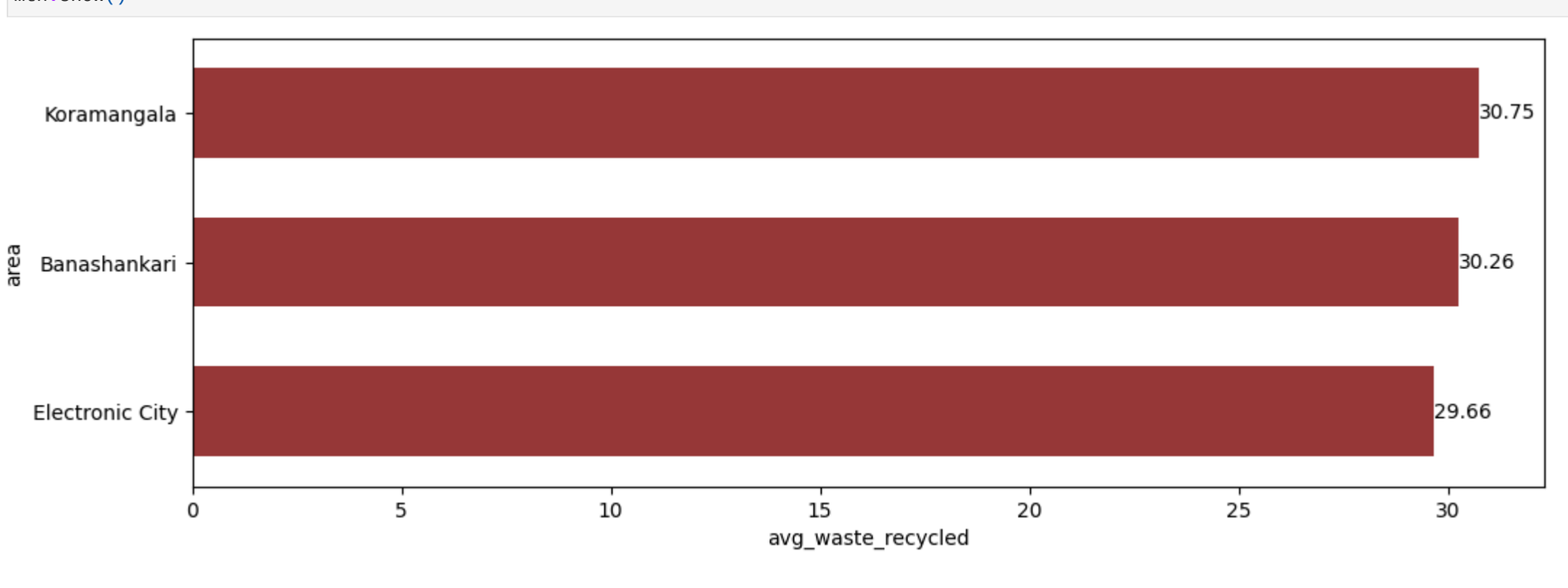
14. Retrieve the top 3 buildings with the highest waste recycled percentage in each city.

In [14]:
```python
query = """ select area, round(avg(cast(waste_recycled_percentage AS decimal)),2) as avg_waste_recycled
from buildings
group by 1
order by 2 desc
limit 3"""
cur.execute(query)
data = cur.fetchall()
df = pd.DataFrame(data, columns = (['area', 'avg_waste_recycled']))
df
```

Out[14]:

	area	avg_waste_recycled
0	Koramangala	30.75
1	Banashankari	30.26
2	Electronic City	29.66

In [23]:
```python
mtn.figure(figsize = (12,4))
ax = sea.barplot( y = 'area', x = 'avg_waste_recycled', data = df, color = 'brown', width = 0.6)
ax.bar_label(ax.containers[0])
mtn.show()
```



15. Find the buildings where the energy consumption per square meter is above 200, but the number of floors is less than or equal to 5.

In [25]:
```python
query = """ select * from
(select area, sum(number_of_floors) as Total_floors,
round(cast(energy_consumption_per_sqm as decimal),2) as energy_consumption
from buildings
group by 1,3)as q
where total_floors <= 5
and energy_consumption > 200"""
cur.execute(query)
data = cur.fetchall()
df = pd.DataFrame(data, columns = (['Area', 'Total_floors', 'Energy_Consumption']))
df
```

Out[25]:

	Area	Total_floors	Energy_Consumption
0	HSR Layout	1	220.72
1	BTM Layout	4	233.51
2	Electronic City	5	224.77
3	Indiranagar	2	231.51
4	HSR Layout	4	206.43
5	BTM Layout	1	224.38
6	HSR Layout	4	245.60
7	Banashankari	5	200.63
8	Electronic City	5	204.10
9	HSR Layout	5	247.72
10	Whitefield	3	233.10
11	Koramangala	5	239.50
12	HSR Layout	1	212.86
13	Hebbal	3	211.58
14	Hebbal	3	248.73
15	Koramangala	4	202.71
16	HSR Layout	2	224.10
17	Koramangala	3	223.96
18	Whitefield	5	248.53
19	Hebbal	1	209.58
20	Banashankari	3	219.33
21	Banashankari	3	211.29
22	Hebbal	3	220.66
23	Yelahanka	2	206.35
24	Koramangala	3	207.03

In [ ]: