In [1]:
```
import pandas as pd
```

In [2]:
```
df = pd.read_csv("Crime_Data.csv")
df.head()
```

Out[2]:

	DATEEND	TIMESTART	TIMEEND	ADDRESS	CODE_DEFINED	Attempt	Arrest	Larcer
0	2020/01/05 05:00:00+00	829	829	200 N STATE ST	LARCENY		NaN	
1	2020/02/14 05:00:00+00	1000	1000	1800 MIDLAND AV	LARCENY		NaN	A
2	2020/01/06 05:00:00+00	1731	1731	500 DELAWARE ST	RAPE		Yes	
3	2020/02/14 05:00:00+00	1933	1933	1 DESTINY USA DR	LARCENY		Yes	A
4	2020/02/14 05:00:00+00	0	1830	400 CHINOOK DR	LARCENY		NaN	A

In [3]:
```
# Shape of the dataset
df.shape

# Column names
df.columns

# Basic info
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 5266 entries, 0 to 5265
Data columns (total 9 columns):
 #   Column        Non-Null Count  Dtype 
---  ------        --------------  ----- 
 0   DATEEND       5266 non-null   object
 1   TIMESTART     5266 non-null   int64 
 2   TIMEEND       5266 non-null   int64 
 3   ADDRESS       5266 non-null   object
 4   CODE_DEFINED  5266 non-null   object
 5   Attempt       5266 non-null   object
 6   Arrest        1262 non-null   object
 7   LarcenyCode   3977 non-null   object
 8   FID           5266 non-null   int64 
dtypes: int64(3), object(6)
memory usage: 370.4+ KB
```

In [4]:
```
# Check missing values in each column
missing_values = df.isnull().sum()
```

```
missing_values
```

Out[4]:
```
DATEEND            0
TIMESTART          0
TIMEEND            0
ADDRESS            0
CODE_DEFINED       0
Attempt            0
Arrest          4004
LarcenyCode     1289
FID                0
dtype: int64
```

In [5]:
```
# Percentage of missing values
(df.isnull().mean() * 100).round(2)
```

Out[5]:
```
DATEEND          0.00
TIMESTART        0.00
TIMEEND          0.00
ADDRESS          0.00
CODE_DEFINED     0.00
Attempt          0.00
Arrest          76.03
LarcenyCode     24.48
FID              0.00
dtype: float64
```

In [6]:
```
df.columns
```

Out[6]:
```
Index(['DATEEND', 'TIMESTART', 'TIMEEND', 'ADDRESS', 'CODE_DEFINED', 'Attempt',
       'Arrest', 'LarcenyCode', 'FID'],
      dtype='object')
```

In [7]:
```
# Convert DATEEND to datetime
df['DATEEND'] = pd.to_datetime(df['DATEEND'], errors='coerce')
```

In [8]:
```
df['DATEEND'].head()
```

Out[8]:
```
0   2020-01-05 05:00:00+00:00
1   2020-02-14 05:00:00+00:00
2   2020-01-06 05:00:00+00:00
3   2020-02-14 05:00:00+00:00
4   2020-02-14 05:00:00+00:00
Name: DATEEND, dtype: datetime64[ns, UTC]
```

In [9]:
```
df['Year'] = df['DATEEND'].dt.year
df['Month'] = df['DATEEND'].dt.month
df['Day'] = df['DATEEND'].dt.day
df['Day_of_Week'] = df['DATEEND'].dt.day_name()
```

In [10]:
```
df[['DATEEND', 'Year', 'Month', 'Day_of_Week']].head()
```

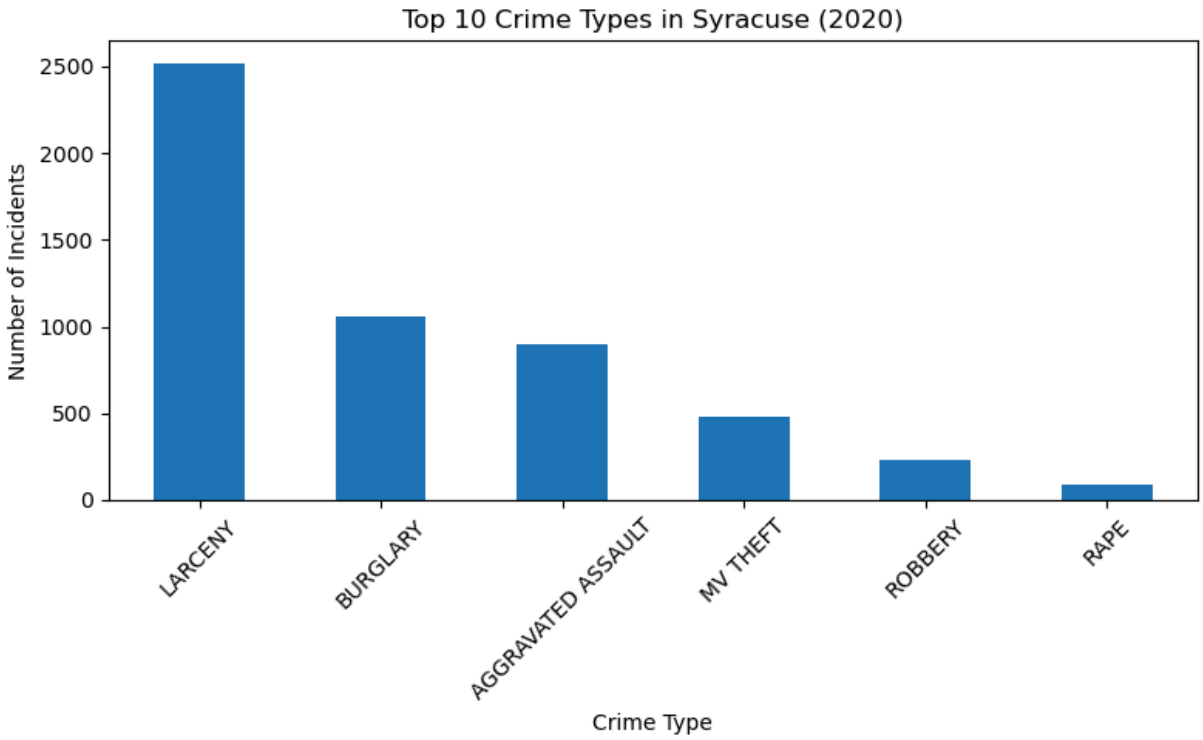
Out[10]:

	DATEEND	Year	Month	Day_of_Week
0	2020-01-05 05:00:00+00:00	2020	1	Sunday
1	2020-02-14 05:00:00+00:00	2020	2	Friday
2	2020-01-06 05:00:00+00:00	2020	1	Monday
3	2020-02-14 05:00:00+00:00	2020	2	Friday
4	2020-02-14 05:00:00+00:00	2020	2	Friday

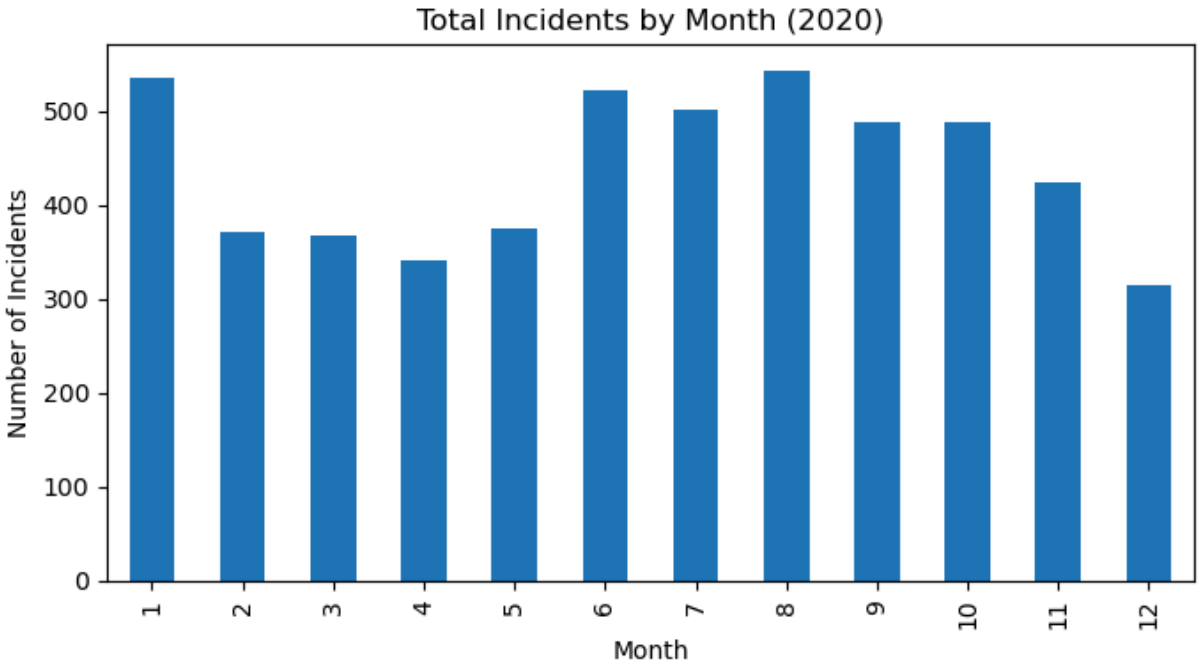
In [11]:
```
offense_counts = df['CODE_DEFINED'].value_counts()
offense_counts.head()
```

Out[11]:
```
CODE_DEFINED
LARCENY                2522
BURGLARY               1054
AGGRAVATED ASSAULT      894
MV THEFT                480
ROBBERY                 227
Name: count, dtype: int64
```

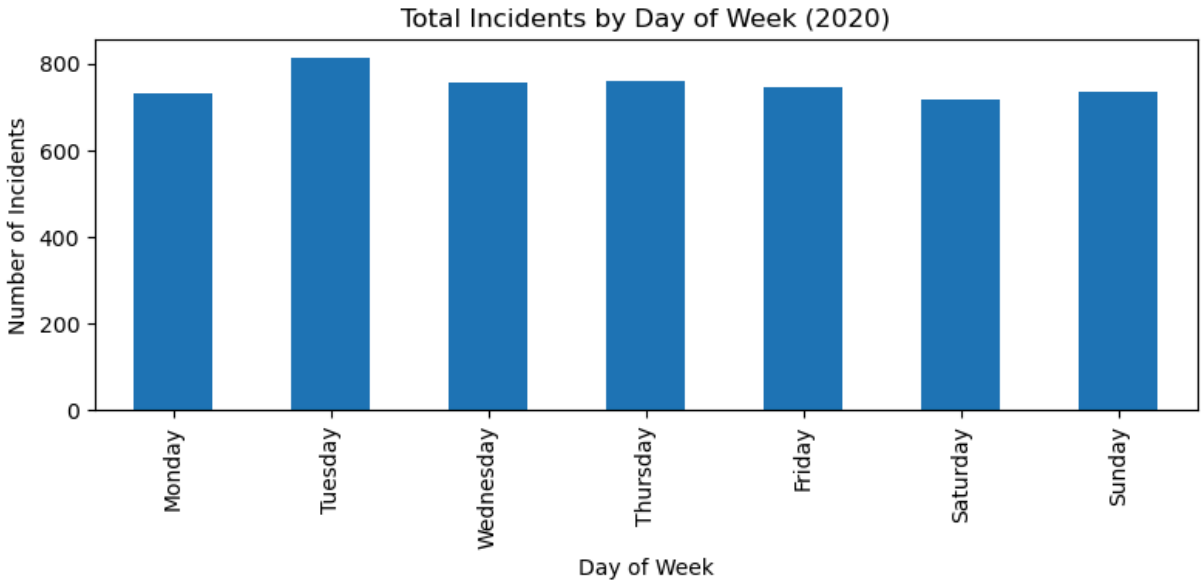
In [12]:
```
import matplotlib.pyplot as plt

offense_counts.head(10).plot(kind='bar', figsize=(8,5))
plt.title("Top 10 Crime Types in Syracuse (2020)")
plt.xlabel("Crime Type")
plt.ylabel("Number of Incidents")
plt.xticks(rotation=45)
plt.tight_layout()
plt.show()
```



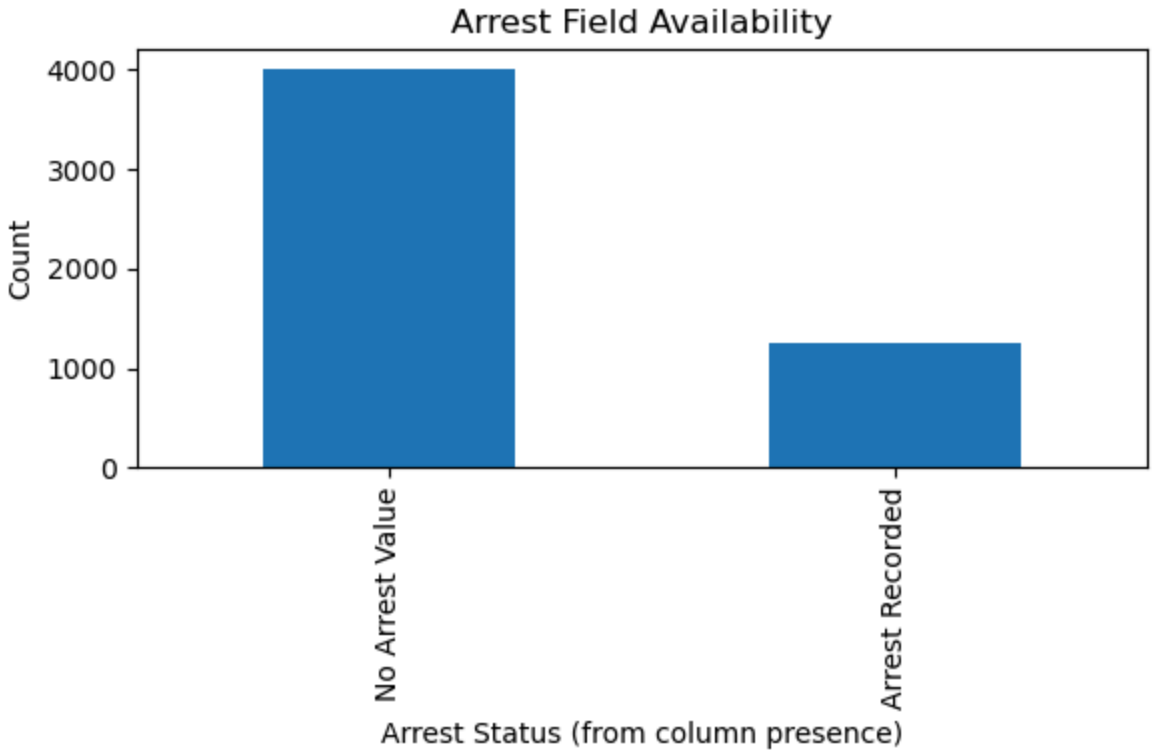
In [13]:
```
month_counts = df['Month'].value_counts().sort_index()

month_counts.plot(kind='bar', figsize=(7,4))
plt.title("Total Incidents by Month (2020)")
plt.xlabel("Month")
plt.ylabel("Number of Incidents")
plt.tight_layout()
plt.show()
```



In [14]:
```
order = ['Monday','Tuesday','Wednesday','Thursday','Friday','Saturday','Sunday']
dow_counts = df['Day_of_Week'].value_counts().reindex(order)

dow_counts.plot(kind='bar', figsize=(8,4))
plt.title("Total Incidents by Day of Week (2020)")
plt.xlabel("Day of Week")
plt.ylabel("Number of Incidents")
plt.tight_layout()
plt.show()
```

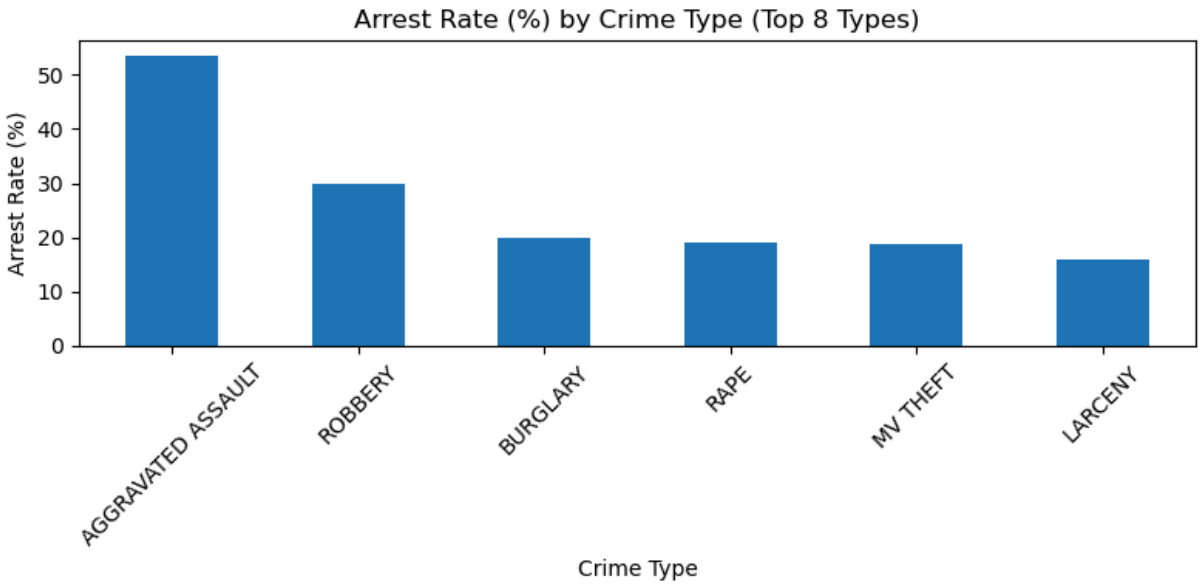


In [17]:
```
arrest_flag = df['Arrest'].notna().map({True: "Arrest Recorded", False: "No Arrest
arrest_counts = arrest_flag.value_counts()

arrest_counts.plot(kind='bar', figsize=(6,4))
plt.title("Arrest Field Availability")
plt.xlabel("Arrest Status (from column presence)")
plt.ylabel("Count")
plt.tight_layout()
plt.show()
```



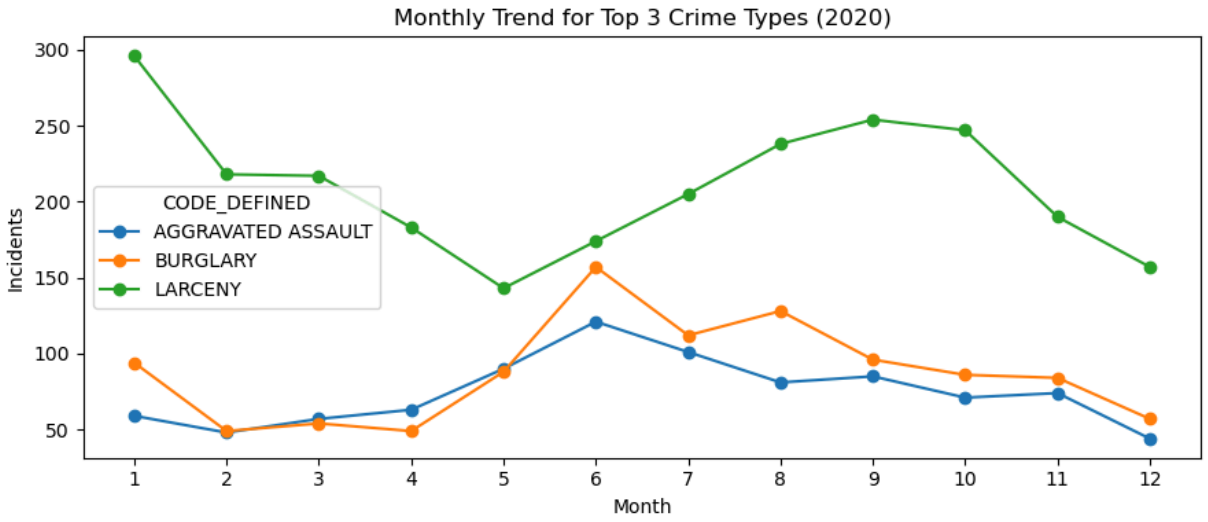
In [18]:
```
df['ArrestFlag'] = df['Arrest'].notna()

top_types = df['CODE_DEFINED'].value_counts().head(8).index
```

```
tmp = df[df['CODE_DEFINED'].isin(top_types)].copy()

arrest_rate = tmp.groupby('CODE_DEFINED')['ArrestFlag'].mean().sort_values(ascendin

(arrest_rate * 100).plot(kind='bar', figsize=(8,4))
plt.title("Arrest Rate (%) by Crime Type (Top 8 Types)")
plt.xlabel("Crime Type")
plt.ylabel("Arrest Rate (%)")
plt.xticks(rotation=45)
plt.tight_layout()
plt.show()
```

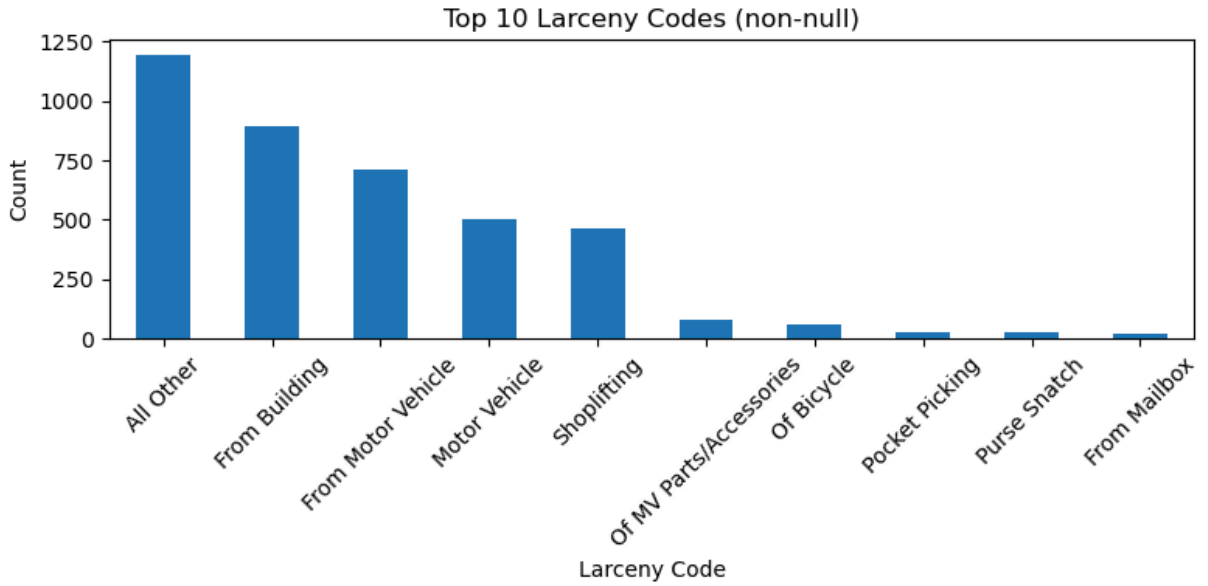


In [19]:

```
top3 = df['CODE_DEFINED'].value_counts().head(3).index
trend = (
    df[df['CODE_DEFINED'].isin(top3)]
    .groupby(['Month','CODE_DEFINED'])
    .size()
    .unstack(fill_value=0)
    .sort_index()
)

trend.plot(kind='line', marker='o', figsize=(9,4))
plt.title("Monthly Trend for Top 3 Crime Types (2020)")
plt.xlabel("Month")
plt.ylabel("Incidents")
plt.xticks(range(1,13))
plt.tight_layout()
plt.show()
```



In [20]:
```
larceny_code_counts = df['LarcenyCode'].dropna().value_counts().head(10)

larceny_code_counts.plot(kind='bar', figsize=(8,4))
plt.title("Top 10 Larceny Codes (non-null)")
plt.xlabel("Larceny Code")
plt.ylabel("Count")
plt.xticks(rotation=45)
plt.tight_layout()
plt.show()
```



In [ ]:
```
```