In [6]:

```
#overall means for all activities for 1991/92
import pandas as pd
import matplotlib.pyplot as plt


winterActiv2 = pd.read_csv(r'C:/Users/HP/Documents/python/1991_92_11_activities.csv')
p91 = winterActiv2.mean()
rounded_p91 = (100 * p91).round().astype(int)

print(rounded_p91)
```

```
alpine skiing           71
cross-country skiing    18
ski touring              9
ice-skating              6
sleigh riding           16
hiking                  30
relaxing                51
shopping                25
sight-seeing            11
museums                  6
pool/sauna              30
dtype: int32
```

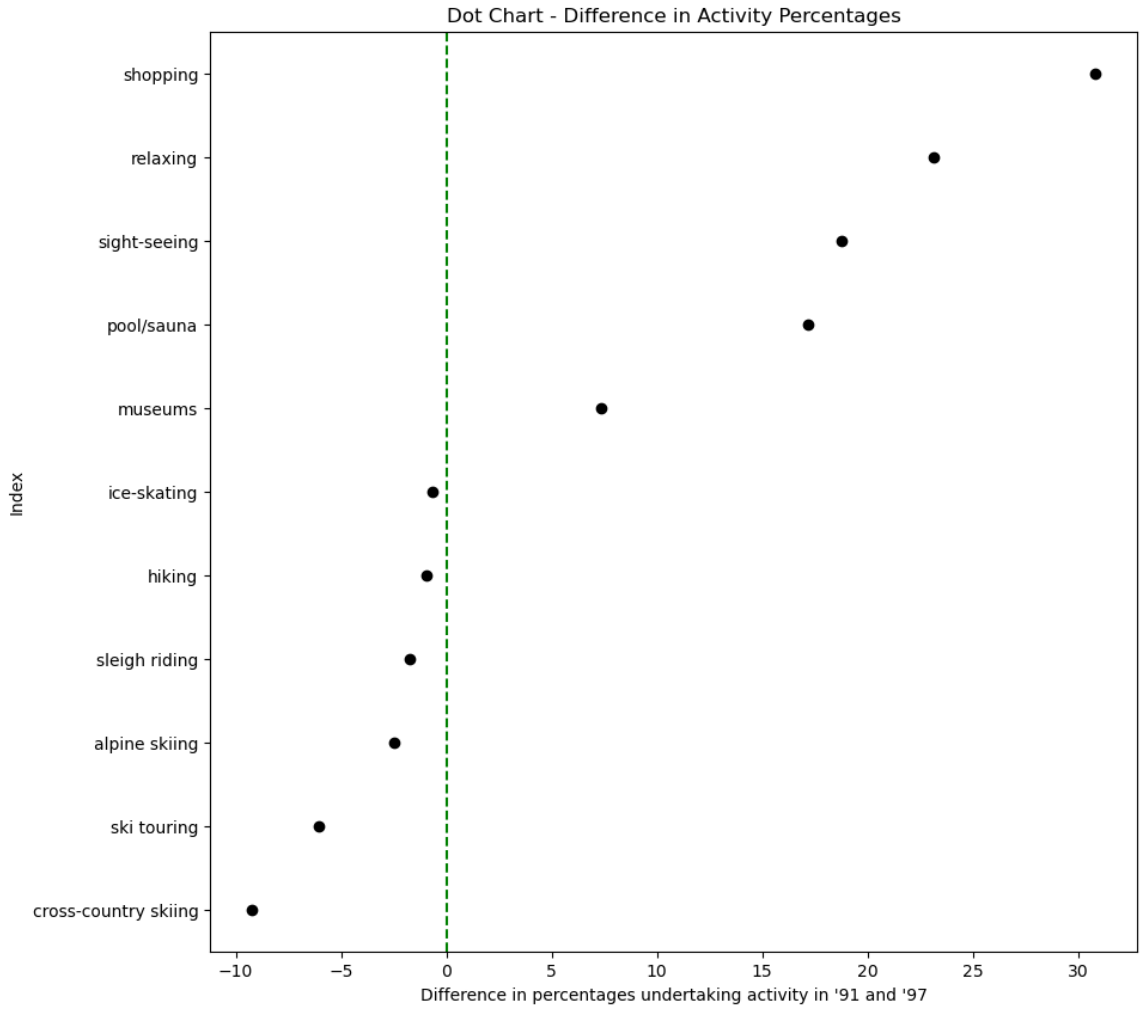
In [7]:

```
#overall means for all activities for 1997/98

winterActiv2 = pd.read_csv(r'C:/Users/HP/Documents/python/1997_98_11_activities.csv')
p97 = winterActiv2.mean()
rounded_p97 = (100 * p97).round().astype(int)

print(rounded_p97)
```

```
alpine skiing           68
cross-country skiing     9
ski touring              3
ice-skating              5
sleigh riding           14
hiking                  29
relaxing                74
shopping                55
sight-seeing            30
museums                 14
pool/sauna              47
dtype: int32
```

In [6]:

```
import pandas as pd
import matplotlib.pyplot as plt

winterActiv2 = pd.read_csv(r'C:/Users/HP/Documents/python/1991_92_11_activities.csv')
winterActiv3 = pd.read_csv(r'C:/Users/HP/Documents/python/1997_98_11_activities.csv')

p91 = winterActiv2.mean()
p97 = winterActiv3.mean()

difference = (p97 - p91).sort_values() * 100

plt.figure(figsize=(10, 10))
plt.plot(difference, range(len(difference)), 'ko')
plt.axvline(0, linestyle='--', color='green')
plt.xlabel("Difference in percentages undertaking activity in '91 and '97")
plt.ylabel("Index")
plt.title("Dot Chart - Difference in Activity Percentages")

# Add activity names as y-axis labels
plt.yticks(range(len(difference)), difference.index)

plt.show()
```

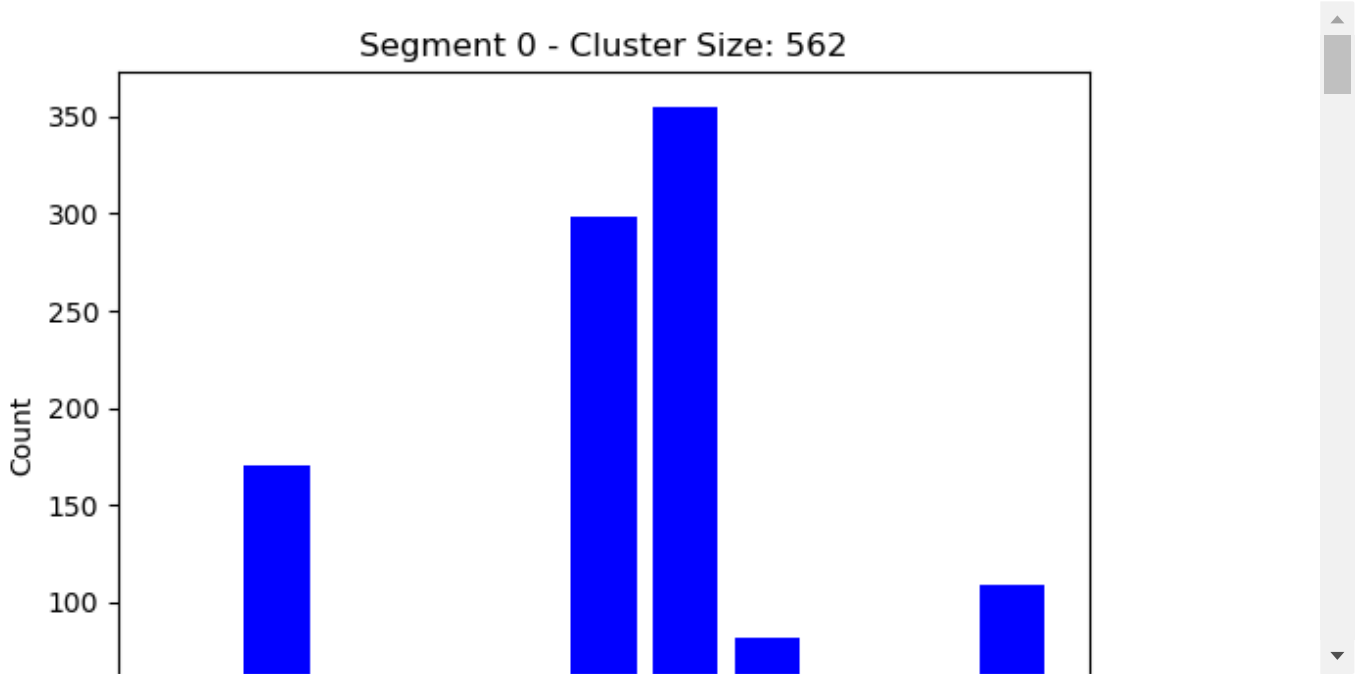


In [7]:

```
import pandas as pd
from sklearn.cluster import KMeans

# Define your dataset or read it from a file
wi91act = pd.read_csv(r'C:/Users/HP/Documents/python/1991_92_11_activities.csv')

# Set the random seed for reproducibility
np.random.seed(1234)

k = 6  # Number of clusters
nrep = 20  # Number of repetitions

kmeans = KMeans(n_clusters=k, n_init=nrep, random_state=1234)
kmeans.fit(wi91act)

wi91act_k6 = kmeans.labels_
```

In [8]:

```
import matplotlib.pyplot as plt
import pandas as pd
from sklearn.cluster import KMeans

# Assuming 'wi91act' is a pandas DataFrame containing the data for 1991
k = 6
nrep = 20
kmeans = KMeans(n_clusters=k, random_state=123456, n_init=nrep)
kmeans.fit(wi91act)
wi91act_k6 = kmeans


# Assuming you have the data for 1991 in a CSV file named 'wi91act.csv'
wi91act = pd.read_csv(r'C:/Users/HP/Documents/python/1991_92_11_activities.csv')

# Assuming 'wi91act_k6' is the cluster solution object obtained from previous code
# Assuming 'wi91act' is a pandas DataFrame containing the data for 1991

# Get the column names of the DataFrame
column_names = wi91act.columns

# Define the cluster labels and their counts
labels = wi91act_k6.labels_
counts = pd.Series(labels).value_counts().sort_index()

# Iterate over each segment
for segment in counts.index:
    segment_counts = counts[segment]

    # Get the column counts for the segment
    segment_column_counts = wi91act.loc[labels == segment].sum(axis=0)

    # Create a bar chart for the segment
    plt.bar(column_names, segment_column_counts, color='blue')

    # Customize the chart
    plt.xlabel('Column')
    plt.ylabel('Count')
    plt.title(f'Segment {segment} - Cluster Size: {segment_counts}')
    plt.xticks(rotation='vertical')

    # Display the chart
    plt.show()
```



In [14]:

```
# Assuming 'wi91act_k6' is the cluster solution object obtained from previous code
# Assuming 'wi91act' and 'wi97act' are pandas DataFrames containing the data for 1991 and
#define wi91act and wi97act
wi91act = pd.read_csv(r'C:/Users/HP/Documents/python/1991_92_11_activities.csv')
wi97act = pd.read_csv(r'C:/Users/HP/Documents/python/1997_98_11_activities.csv')

# Calculate cluster sizes for 1991
labels91 = wi91act_k6.labels_
size91 = pd.Series(labels91).value_counts().sort_index()

# Calculate cluster sizes for 1997
labels97 = wi91act_k6.predict(wi97act)
size97 = pd.Series(labels97).value_counts().sort_index()

# Combine size91 and size97 into a single table
cluster_sizes = pd.concat([size91, size97], axis=1)
cluster_sizes.columns = ['1991', '1997']

# Calculate the percentage and round to the nearest whole number
cluster_sizes_percentage = round(cluster_sizes.apply(lambda x: x / x.sum() * 100, axis=1)

# Display the cluster sizes as percentages
print(cluster_sizes_percentage)
```

```
   1991  1997
0  72.0  28.0
1  31.0  69.0
2  80.0  20.0
3  29.0  71.0
4  50.0  50.0
5  62.0  38.0
```

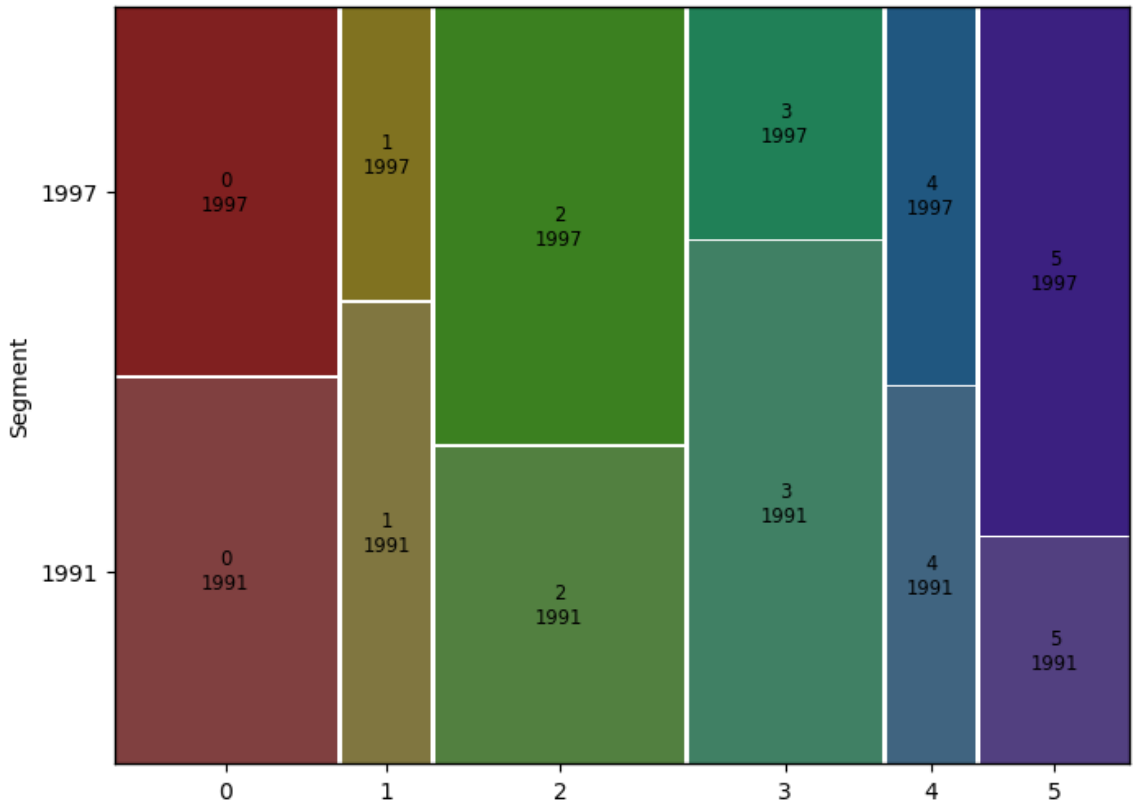
In [15]:

```
import numpy as np
from scipy.stats import chi2_contingency

size91 = np.array([23, 11, 21, 27, 9, 9])
size97 = np.array([22, 7, 29, 12, 9, 21])

observed = np.array([size91, size97])

chi2, p, dof, expected = chi2_contingency(observed)

print(f"Chi-square statistic: {chi2}")
print(f"P-value: {p}")
print(f"Degrees of freedom: {dof}")
print("Expected frequencies:")
print(expected)
```

```
Chi-square statistic: 12.76034188034188
P-value: 0.025731179610575288
Degrees of freedom: 5
Expected frequencies:
[[22.5  9.  25.  19.5  9.  15. ]
 [22.5  9.  25.  19.5  9.  15. ]]
```

In [17]:

```
import pandas as pd
import matplotlib.pyplot as plt
from statsmodels.graphics.mosaicplot import mosaic

# Assuming 'size91' and 'size97' are the cluster sizes obtained previously

# Combine the cluster sizes into a DataFrame
size_data = pd.DataFrame({"1991": size91, "1997": size97})

# Specify the index for the DataFrame
size_data.index.name = "Segment"

# Create a mosaic plot
fig, ax = plt.subplots(figsize=(8, 6))
mosaic(size_data.stack(), ax=ax, title="")

# Customize the plot
ax.set_ylabel("Segment")

# Show the plot
plt.show()
```

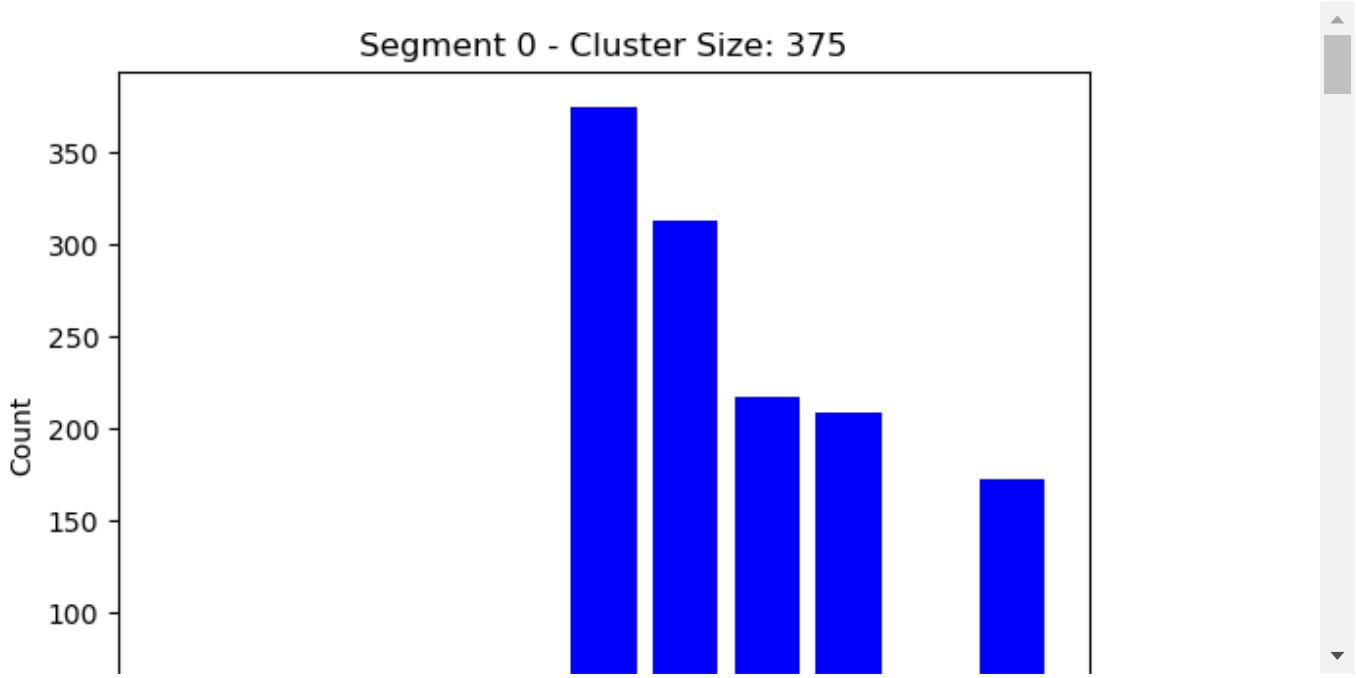


In [21]:

```
#from sklearn.cluster import KMeans

# Assuming 'wi97act' is a pandas DataFrame containing the data for 1997
# Assuming 'wi91act_k6' is the cluster solution object obtained from the 1991 data

# Get the number of clusters from the 1991 cluster solution
k = wi97act_k6.n_clusters

# Set the value of n_init explicitly to suppress the warning
n_init = 10

# Perform clustering for 1997
kmeans_97 = KMeans(n_clusters=k, random_state=1234, n_init=n_init)
kmeans_97.fit(wi97act)
wi97act_k6 = kmeans_97
```

In [23]:

```
import matplotlib.pyplot as plt
import pandas as pd
from sklearn.cluster import KMeans

# Assuming 'wi91act' is a pandas DataFrame containing the data for 1991
k = 6
nrep = 20
kmeans = KMeans(n_clusters=k, random_state=123456, n_init=nrep)
kmeans.fit(wi91act)
wi71act_k6 = kmeans


# Assuming you have the data for 1991 in a CSV file named 'wi91act.csv'
wi97act = pd.read_csv(r'C:/Users/HP/Documents/python/1997_98_11_activities.csv')

# Assuming 'wi91act_k6' is the cluster solution object obtained from previous code
# Assuming 'wi91act' is a pandas DataFrame containing the data for 1991

# Get the column names of the DataFrame
column_names = wi97act.columns

# Define the cluster labels and their counts
labels = wi97act_k6.labels_
counts = pd.Series(labels).value_counts().sort_index()

# Iterate over each segment
for segment in counts.index:
    segment_counts = counts[segment]

    # Get the column counts for the segment
    segment_column_counts = wi97act.loc[labels == segment].sum(axis=0)

    # Create a bar chart for the segment
    plt.bar(column_names, segment_column_counts, color='blue')

    # Customize the chart
    plt.xlabel('Column')
    plt.ylabel('Count')
    plt.title(f'Segment {segment} - Cluster Size: {segment_counts}')
    plt.xticks(rotation='vertical')

    # Display the chart
    plt.show()
```



In [ ]: