In [26]:

```
#

import pandas as pd
import matplotlib.pyplot as plt

file_path = './Downloads/bayview.xlsx'
df = pd.read_excel(file_path)
print(df)
```

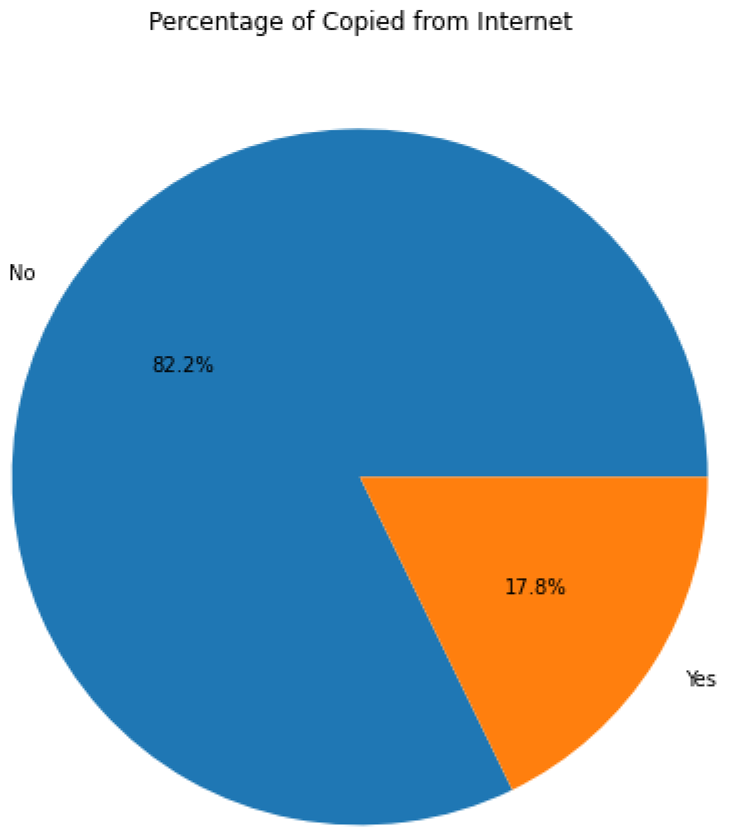
```
    Student Copied from Internet Copied on Exam  \
0         1                   No             No   
1         2                   No             No   
2         3                  Yes             No   
3         4                  Yes            Yes   
4         5                   No             No   
..      ...                  ...            ...   
85       86                   No             No   
86       87                   No             No   
87       88                   No             No   
88       89                   No            Yes   
89       90                   No             No   

   Collaborated on Individual Project  Gender  
0                                  No  Female  
1                                  No    Male  
2                                 Yes    Male  
3                                  No    Male  
4                                 Yes    Male  
..                                ...     ...  
85                                 No    Male  
86                                Yes    Male  
87                                 No    Male  
88                                Yes    Male  
89                                 No  Female  

[90 rows x 5 columns]
```

In [27]:

```
col1_value_counts = df['Copied from Internet'].value_counts()/90
plt.figure(figsize=(8, 8))
plt.pie(col1_value_counts, labels=col1_value_counts.index, autopct='%1.1
plt.title('Percentage of Copied from Internet')
```

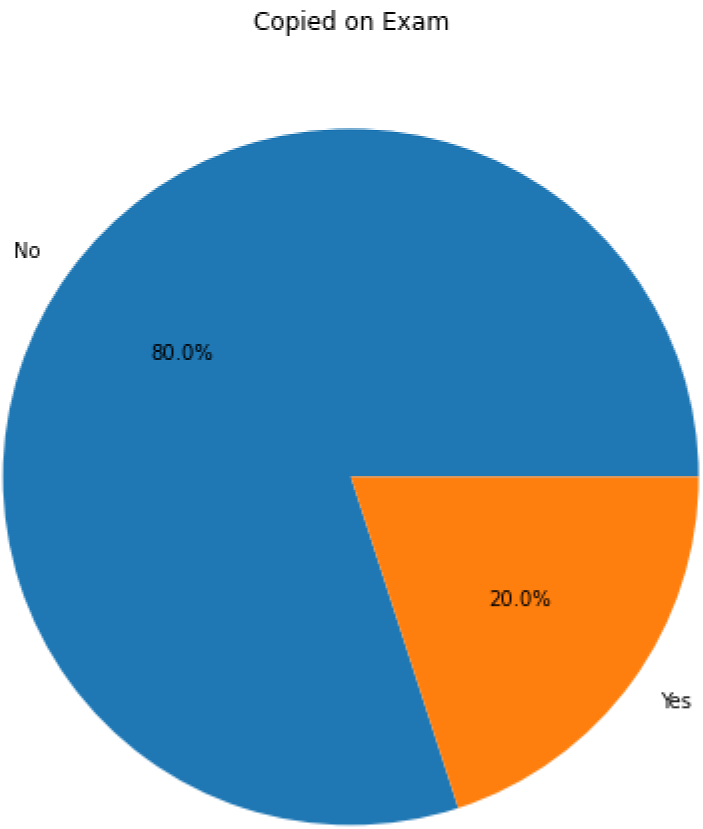
Out[27]:

```
Text(0.5, 1.0, 'Percentage of Copied from Internet')
```



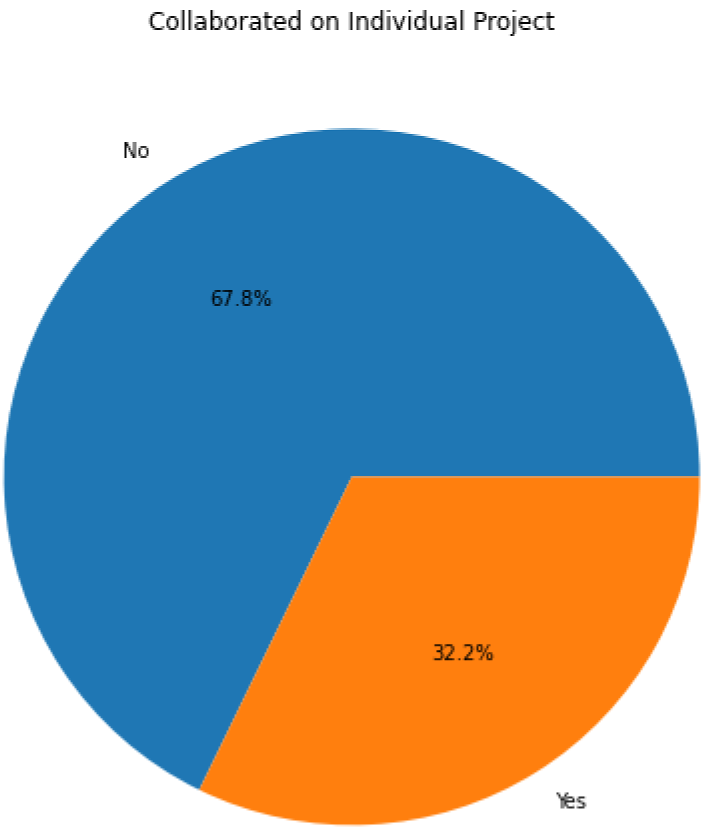
In [28]:
```
col2_value_counts = df['Copied on Exam'].value_counts()/90
plt.figure(figsize=(8, 8))
plt.pie(col2_value_counts, labels=col2_value_counts.index, autopct='%1.1
plt.title('Copied on Exam')
```

Out[28]:
```
Text(0.5, 1.0, 'Copied on Exam')
```



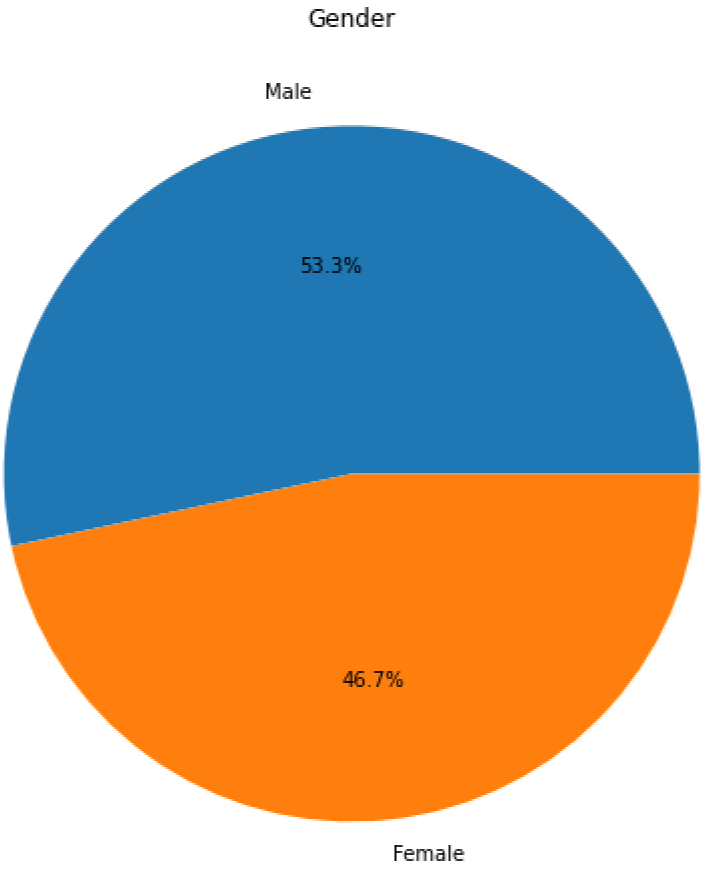
In [29]:
```
col3_value_counts = df['Collaborated on Individual Project'].value_count
plt.figure(figsize=(8, 8))
plt.pie(col3_value_counts, labels=col3_value_counts.index, autopct='%1.1
plt.title('Collaborated on Individual Project')
```

Out[29]:
```
Text(0.5, 1.0, 'Collaborated on Individual Project')
```



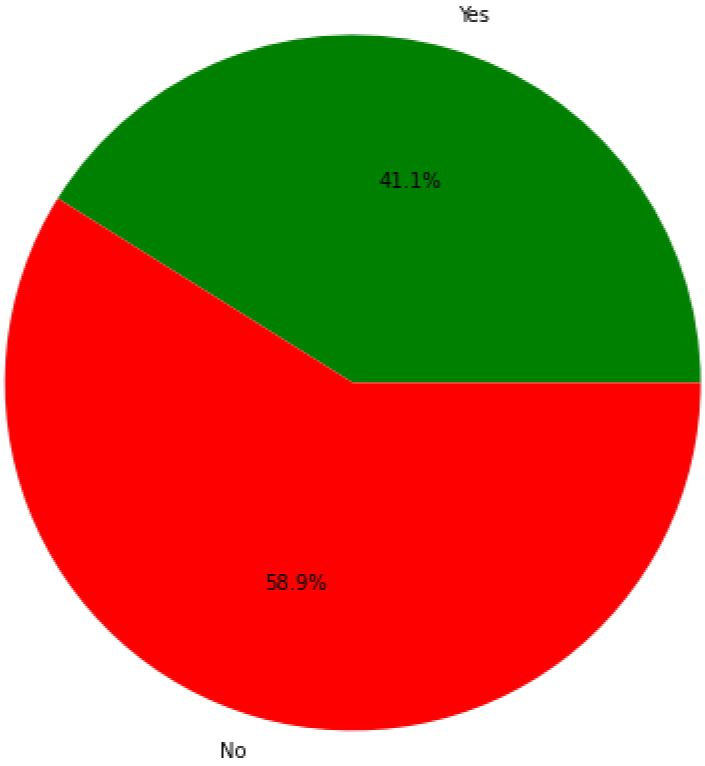
```
In [30]: col4_value_counts = df['Gender'].value_counts()/90
         plt.figure(figsize=(8, 8))
         plt.pie(col4_value_counts, labels=col4_value_counts.index, autopct='%1.1
         plt.title('Gender')
```

```
Out[30]: Text(0.5, 1.0, 'Gender')
```



In [31]:

```
yes_mask = ((df['Collaborated on Individual Project'] == 'Yes') |
            (df['Copied on Exam'] == 'Yes') |
            (df['Copied from Internet'] == 'Yes'))
yes = yes_mask.sum()
no = len(df) - yes
plt.figure(figsize=(8, 8))  # Set the figure size
plt.pie([yes, no], labels=['Yes', 'No'], autopct='%1.1f%%', colors=['gre
plt.show()
```

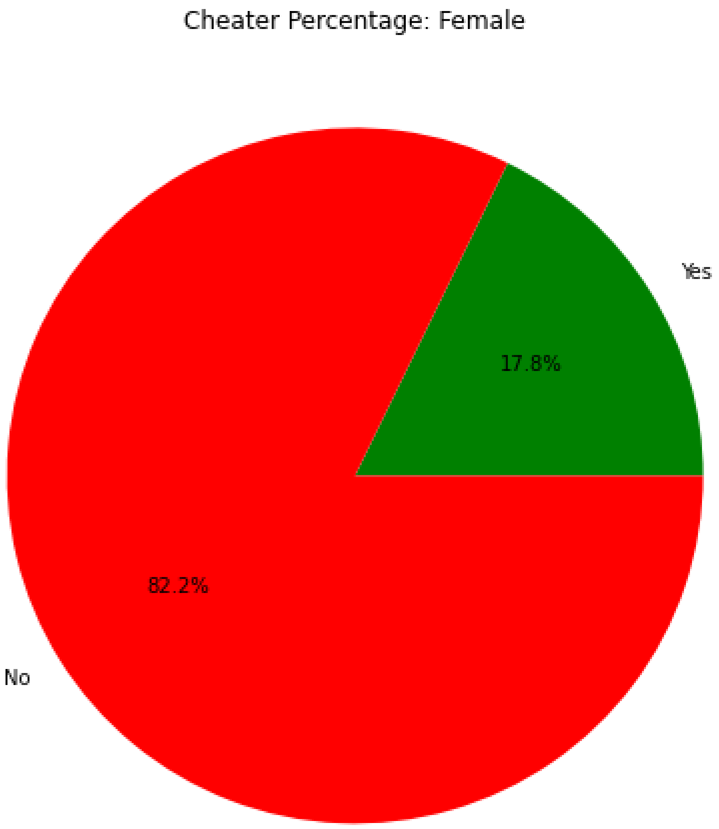


In [32]:
```
heated_In_Multiple = 0
heated_In_One = 0

or index, row in df.iterrows():
   if row['Copied from Internet'] == 'Yes' or row['Copied on Exam'] == 'Y
       if row['Copied from Internet'] == 'Yes' and row['Copied on Exam']
           Cheated_In_Multiple += 1
       elif row['Copied on Exam'] == 'Yes' and row['Collaborated on Indiv
           Cheated_In_Multiple += 1
       elif row['Copied from Internet'] == 'Yes' and row['Copied on Exam'
           Cheated_In_Multiple += 1
       else:
           Cheated_In_One += 1

otal = Cheated_In_One + Cheated_In_Multiple
ercent1 = Cheated_In_One/total
ercent2 = Cheated_In_Multiple/total

rint('The students that only cheated in one category makes up ', Percent1
rint('The students that cheated in MORE THAN one category makes up ', Per
```

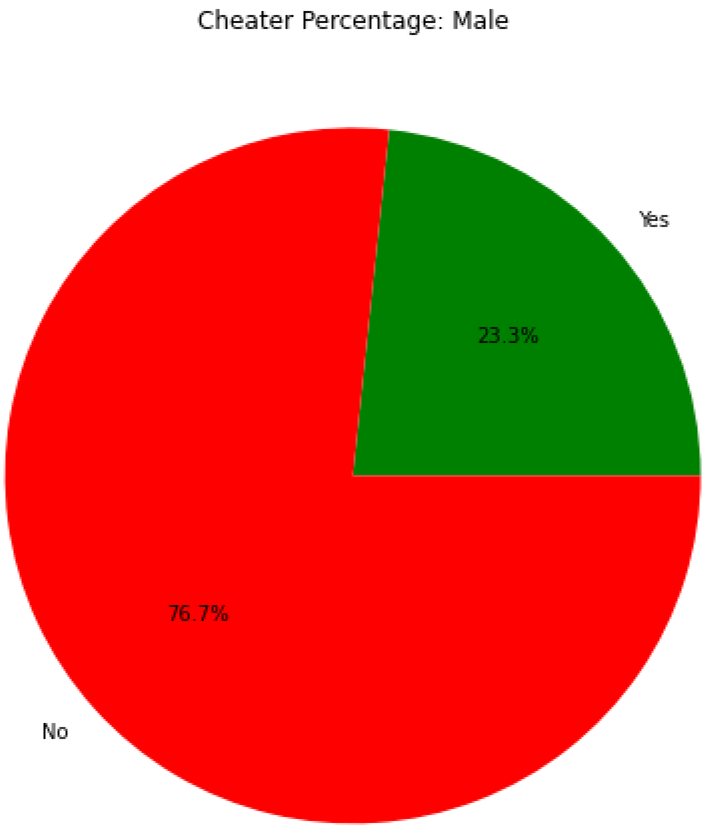
```
The students that only cheated in one category makes up  0.540540540540
5406
The students that cheated in MORE THAN one category makes up  0.4594594
594594595
```

In [33]:
```
df1 = df[(df['Gender'] == 'Female')]
df2 = df[(df['Gender'] == 'Male')]
```

In [34]:
```
yes_mask = ((df1['Collaborated on Individual Project'] == 'Yes') |
            (df1['Copied on Exam'] == 'Yes') |
            (df1['Copied from Internet'] == 'Yes'))
yes = yes_mask.sum()
no = len(df) - yes
plt.figure(figsize=(8, 8))  # Set the figure size
plt.pie([yes, no], labels=['Yes', 'No'], autopct='%1.1f%%', colors=['gre
plt.title('Cheater Percentage: Female')
plt.show()
```



```
In [35]: yes_mask = ((df2['Collaborated on Individual Project'] == 'Yes') |
                     (df2['Copied on Exam'] == 'Yes') |
                     (df2['Copied from Internet'] == 'Yes'))
         yes = yes_mask.sum()
         no = len(df) - yes
         plt.figure(figsize=(8, 8))  # Set the figure size
         plt.pie([yes, no], labels=['Yes', 'No'], autopct='%1.1f%%', colors=['gre
         plt.title('Cheater Percentage: Male')
         plt.show()
```



In [36]:

```
Cheated_In_Multiple = 0
Cheated_In_One = 0

for index, row in df1.iterrows():
    if row['Copied from Internet'] == 'Yes' or row['Copied on Exam'] == '
        if row['Copied from Internet'] == 'Yes' and row['Copied on Exam']
            Cheated_In_Multiple += 1
        elif row['Copied on Exam'] == 'Yes' and row['Collaborated on Indi
            Cheated_In_Multiple += 1
        elif row['Copied from Internet'] == 'Yes' and row['Copied on Exam
            Cheated_In_Multiple += 1
        else:
            Cheated_In_One += 1

total = Cheated_In_One + Cheated_In_Multiple
Percent1 = Cheated_In_One/total
Percent2 = Cheated_In_Multiple/total

print('The students (Female) that only cheated in one category makes up '
print('The students (Female) that cheated in MORE THAN one category makes
```

```
The students (Female) that only cheated in one category makes up  0.5
The students (Female) that cheated in MORE THAN one category makes up
0.5
```

In [37]:

```
ated_In_Multiple = 0
ated_In_One = 0

 index, row in df2.iterrows():
 if row['Copied from Internet'] == 'Yes' or row['Copied on Exam'] == 'Yes
     if row['Copied from Internet'] == 'Yes' and row['Copied on Exam'] ==
         Cheated_In_Multiple += 1
     elif row['Copied on Exam'] == 'Yes' and row['Collaborated on Individ
         Cheated_In_Multiple += 1
     elif row['Copied from Internet'] == 'Yes' and row['Copied on Exam']
         Cheated_In_Multiple += 1
     else:
         Cheated_In_One += 1

al = Cheated_In_One + Cheated_In_Multiple
cent1 = Cheated_In_One/total
cent2 = Cheated_In_Multiple/total

nt('The students (Male) that only cheated in one category makes up ', Per
nt('The students (Male) that cheated in MORE THAN one category makes up '
```

```
The students (Male) that only cheated in one category makes up  0.57142
85714285714
The students (Male) that cheated in MORE THAN one category makes up  0.
42857142857142855
```

In [38]:

```
#95% CI for ALL CHEATERS (YES)
from scipy.stats import norm
import numpy as np

CheaterCount = 0

for index, row in df.iterrows():
    if row['Copied from Internet'] == 'Yes' or row['Copied on Exam'] ==
        CheaterCount += 1

yesProportion = CheaterCount/len(df)
SE = np.sqrt((yesProportion) * (1 - yesProportion)) / (len(df))
ME = norm.ppf(0.975) * SE
CI = (yesProportion - ME, yesProportion + ME)
print(CI)
```

```
(0.400395871946072, 0.4218263502761502)
```

In [39]:

```
#95% CI for ALL CHEATERS (No)

NonCheaterCount = 0

for index, row in df.iterrows():
    if row['Copied from Internet'] == 'No' and row['Copied on Exam'] ==
        NonCheaterCount += 1

noProportion = NonCheaterCount/len(df)
SE = np.sqrt((noProportion) * (1 - noProportion)) / (len(df))
ME = norm.ppf(0.975) * SE
CI = (noProportion - ME, noProportion + ME)
print(CI)
```

```
(0.5781736497238499, 0.5996041280539279)
```

```
In [40]: CI for WOMEN CHEATERS (Yes)


         scipy.stats import norm
         t numpy as np

         erCount = 0

         ndex, row in df1.iterrows():
         f row['Copied from Internet'] == 'Yes' or row['Copied on Exam'] == 'Yes'
            CheaterCount += 1

         oportion = CheaterCount/len(df)
         np.sqrt((yesProportion) * (1 - yesProportion)) / (len(df))
         norm.ppf(0.975) * SE
         (yesProportion - ME, yesProportion + ME)
         (CI)
```

```
(0.16945172952755436, 0.1861038260280012)
```

```
In [41]: 95% CI for WOMEN CHEATERS (No)

         onCheaterCount = 0

         or index, row in df1.iterrows():
            if row['Copied from Internet'] == 'No' and row['Copied on Exam'] == 'N
                NonCheaterCount += 1

         oProportion = NonCheaterCount/len(df)
         E = np.sqrt((noProportion) * (1 - noProportion)) / (len(df))
         E = norm.ppf(0.975) * SE
         I = (noProportion - ME, noProportion + ME)
         rint(CI)
```

```
(0.27901837570063326, 0.29875940207714446)
```

In [42]:

```
#95% CI for MEN CHEATERS (Yes)


from scipy.stats import norm
import numpy as np

CheaterCount = 0

for index, row in df2.iterrows():
    if row['Copied from Internet'] == 'Yes' or row['Copied on Exam'] ==
        CheaterCount += 1

yesProportion = CheaterCount/len(df)
SE = np.sqrt((yesProportion) * (1 - yesProportion)) / (len(df))
ME = norm.ppf(0.975) * SE
CI = (yesProportion - ME, yesProportion + ME)
print(CI)
```

```
(0.2241225351872871, 0.2425441314793796)
```

In [43]:

```
#95% CI for MEN CHEATERS (No)

NonCheaterCount = 0

for index, row in df2.iterrows():
    if row['Copied from Internet'] == 'No' and row['Copied on Exam'] ==
        NonCheaterCount += 1

noProportion = NonCheaterCount/len(df)
SE = np.sqrt((noProportion) * (1 - noProportion)) / (len(df))
ME = norm.ppf(0.975) * SE
CI = (noProportion - ME, noProportion + ME)
print(CI)
```

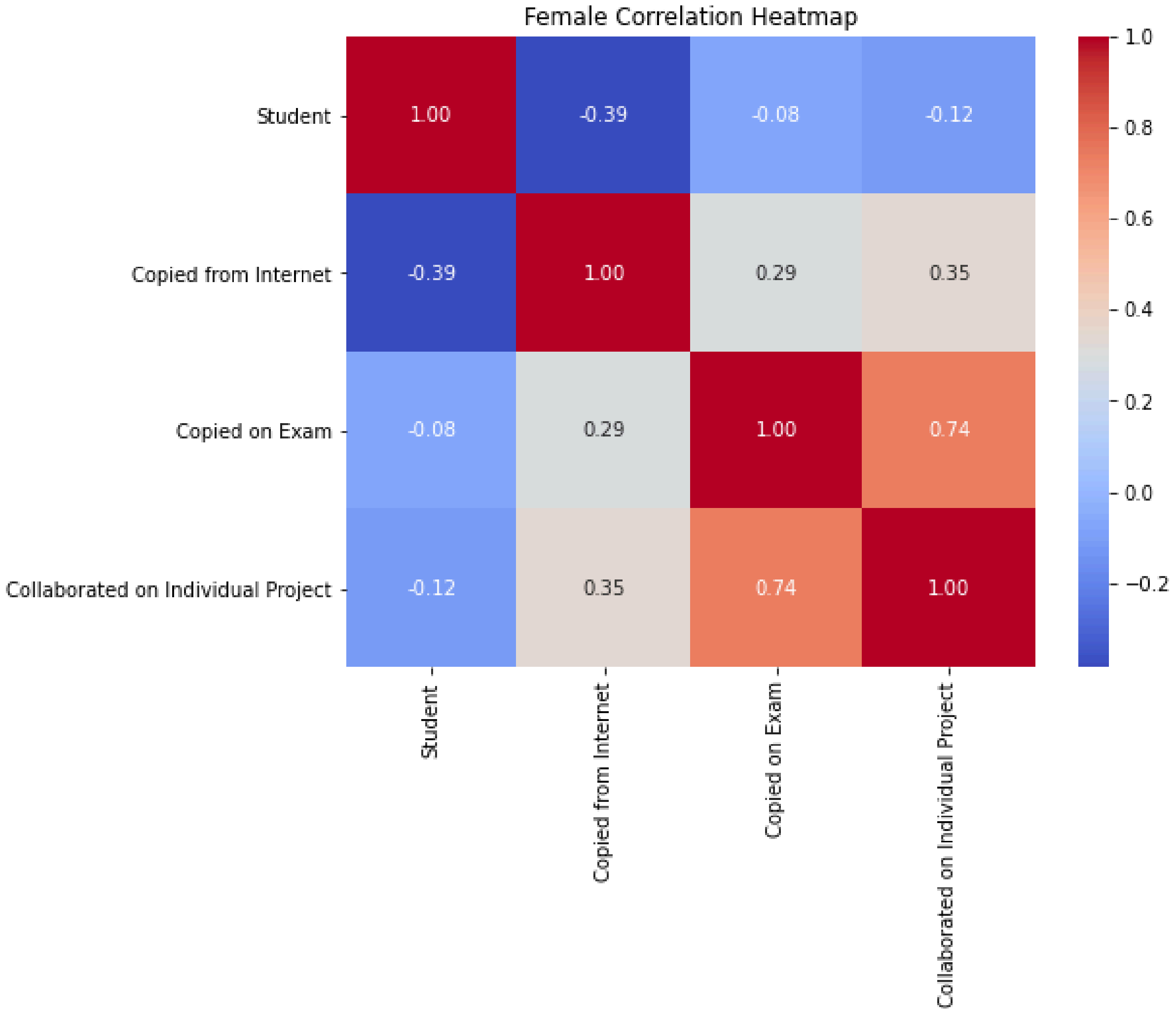
```
(0.2900203518682866, 0.30997964813171336)
```

In [49]:

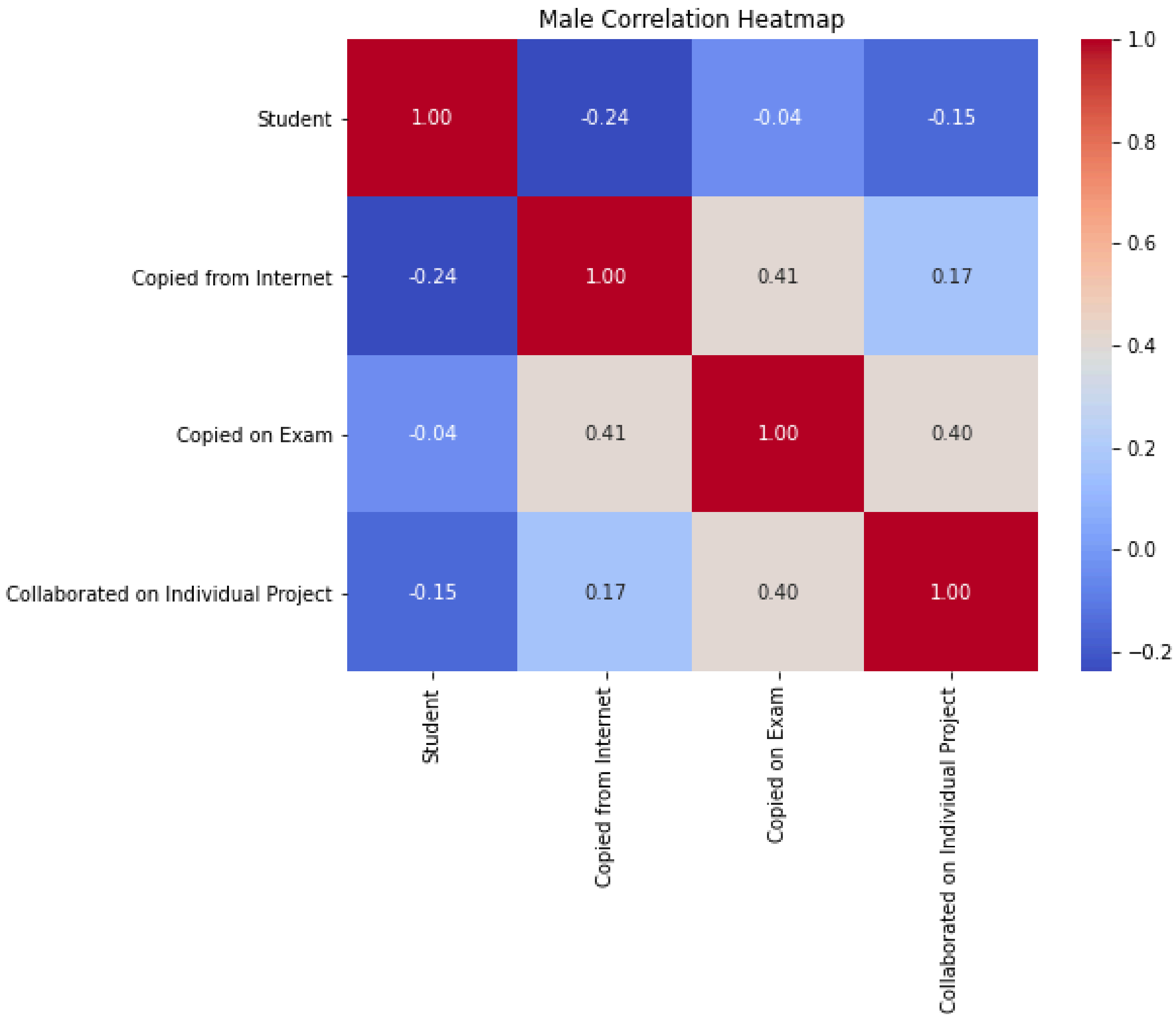
```
df1.replace({'Yes': 1, 'No': 0}, inplace=True)
df2.replace({'Yes': 1, 'No': 0}, inplace=True)
```

```
/opt/anaconda3/lib/python3.8/site-packages/pandas/core/frame.py:4524: S
ettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-
docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy (ht
tps://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#ret
urning-a-view-versus-a-copy)
  return super().replace(
```

In [52]:
```
#Heat Map Women Corr
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt


corr_matrix = df1.corr()

# Plot heatmap
plt.figure(figsize=(8, 6))
sns.heatmap(corr_matrix, annot=True, cmap='coolwarm', fmt=".2f")
plt.title('Female Correlation Heatmap')
plt.show()
```



In [53]:
```
#Heat Map Men Corr


corr_matrix = df2.corr()

# Plot heatmap
plt.figure(figsize=(8, 6))
sns.heatmap(corr_matrix, annot=True, cmap='coolwarm', fmt=".2f")
plt.title('Male Correlation Heatmap')
plt.show()
```



In [54]:
```
#Hypothesis Test 1

#YBar = .411
#H0 = .56
#n = 90
```

In [57]:
```
def check_yes(row):
    if 'Yes' in row.values:
        return 'Yes'
    else:
        return 'No'

# Apply the function row-wise and create a new column
df['Cheated?'] = df.apply(check_yes, axis=1)

df['Cheated?'].value_counts()/90
```

Out[57]:
```
No     0.588889
Yes    0.411111
Name: Cheated?, dtype: float64
```

In [61]:
```
import numpy as np
from scipy.stats import norm

p_hat = 0.411  # Sample proportion
n = len(df)
H0 = 0.56
alpha = 0.05
q = (1 - 0.56)

z_score = (p_hat - H0) / np.sqrt((H0 * (q)) / n)

z_critical = norm.ppf(alpha)
print(z_critical)
print(z_score)

# Make a decision
if z_score < z_critical:
    print("Reject null hypothesis: There is evidence that the population
else:
    print("Fail to reject null hypothesis: There is not enough evidence t
```

```
-1.6448536269514729
-2.847653682157734
Reject null hypothesis: There is evidence that the population proportio
n is less than 0.56
```

In [62]:

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

p_hat = 0.411  # Sample proportion
n = len(df)
H0 = 0.47
alpha = 0.05
q = (1 - 0.47)

z_score = (p_hat - H0) / np.sqrt((H0 * (q)) / n)

z_critical = norm.ppf(alpha)
print(z_critical)
print(z_score)

# Make a decision
if z_score < z_critical:
    print("Reject null hypothesis: There is evidence that the population
else:
    print("Fail to reject null hypothesis: There is not enough evidence
```

```
-1.6448536269514729
-1.12146675190675
Fail to reject null hypothesis: There is not enough evidence to suggest
that the population proportion is less than 0.56
```

In [63]:

```
df['Gender'].value_counts()
```

Out[63]:

```
Male      48
Female    42
Name: Gender, dtype: int64
```

In [ ]: