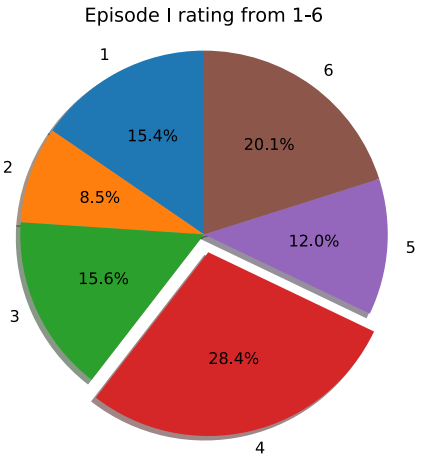
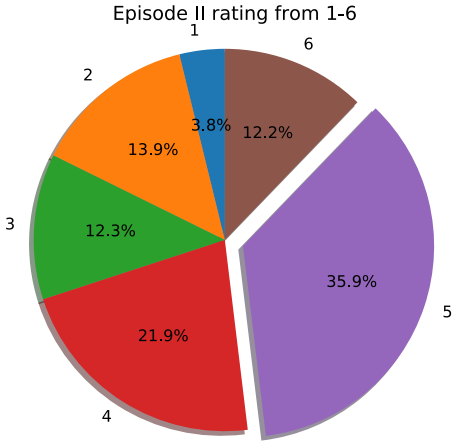
In [1]:
```
#Task 1: Data Preparation
#Task 1.1: Data Retrieving
import pandas as pd

data = pd.read_csv('StarWars.csv',encoding='unicode escape',header=None)
data = data.drop([0,1])

# Index starts from 0

# ID index 0
# Wached Star Wars index 1
# Fan of Star Wars index 2
# Watched episode index 3-8
# Ranking index 9-14
# Character rating index 15-28
# "Who shot first" index 29
# The Expanded Universe index 30
# Fan of the Expanded Universe index 31
# Fan of Star Trek index 32
# Gender index 33
# Age index 34
# Education index 35
# Location index 36

# Give header names to the columns
data.columns = ['Respondent ID',
                'Watched the Star Wars franchise?',
                'Fan of the franchise?',
                'Seen Ep I',
                'Seen Ep II',
                'Seen Ep III',
                'Seen Ep IV',
                'Seen Ep V',
                'Seen Ep VI',
                'Rank The Film From 1-6 for EP I',
                'Rank The Film From 1-6 for EP II',
                'Rank The Film From 1-6 for EP III',
                'Rank The Film From 1-6 for EP IV',
                'Rank The Film From 1-6 for EP V',
                'Rank The Film From 1-6 for EP VI',
                'How do you like the character: Han Solo',
                'How do you like the character: Luke Skywalker',
                'How do you like the character: Princess Leia Organa',
                'How do you like the character: Anakin Skywalker',
                'How do you like the character: Obi Wan Kenobi',
                'How do you like the character: Emperor Palpatine',
                'How do you like the character: Darth Vader',
                'How do you like the character: Lando Calrissian',
                'How do you like the character: Boba Fett',
                'How do you like the character: C-3P0',
                'How do you like the character: R2 D2',
                'How do you like the character: Jar Jar Binks',
                'How do you like the character: Padme Amidala',
                'How do you like the character: Yoda',
                'Which character shot first?',
                'Familiar with the Expanded Universe?',
                'Fan of the Expanded Universe?',
                'Fan of the Star Trek franchise?',
                'Gender',
                'Age',
                'Household Income',
                'Education',
                'Location (Census Region)']

#Task 1.4: Extra-whitespaces (For initial data cleansing - overall str.strip() is also done towards the end)
data['Watched the Star Wars franchise?'] = data['Watched the Star Wars franchise?'].str.strip()
data['Watched the Star Wars franchise?'].fillna('Did not answer', inplace=True)
data['Fan of the franchise?'] = data['Fan of the franchise?'].str.strip()

#Task 1.7: Missing values (This step is done for everywhere that uses .fillna function)
data['Fan of the franchise?'].fillna('Did not answer', inplace=True)

watchedColumns = ["Seen Ep I",
                  "Seen Ep II",
                  "Seen Ep III",
                  "Seen Ep IV",
                  "Seen Ep V",
                  "Seen Ep VI"]

episodes = ['Star Wars: Episode I  The Phantom Menace',
            'Star Wars: Episode II  Attack of the Clones',
            'Star Wars: Episode III  Revenge of the Sith',
            'Star Wars: Episode IV  A New Hope',
            'Star Wars: Episode V The Empire Strikes Back',
            'Star Wars: Episode VI Return of the Jedi']

filmRating = ['Rank The Film From 1-6 for EP I',
              'Rank The Film From 1-6 for EP II',
              'Rank The Film From 1-6 for EP III',
              'Rank The Film From 1-6 for EP IV',
              'Rank The Film From 1-6 for EP V',
              'Rank The Film From 1-6 for EP VI']

observation_index = 2
max_observation_index = len(data.index)

# Replace the text value to a readible and sensible data collection
for col in watchedColumns:
    for value in episodes:
        data[col].replace(value,'Watched',inplace=True)
        data[col].fillna('Not Watched', inplace=True)
```

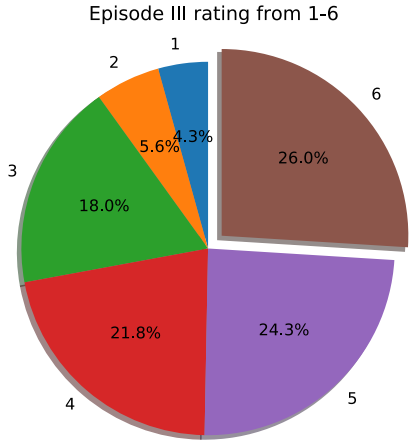
```
# Task 1.7: Filling missing values
for value in filmRating:
    data[value].fillna('0', inplace=True)

# Task 1.6: Sanity checks
# If they haven't seen the Star Wars franchise, they cannot possibly be a fan
while (observation_index <= max_observation_index):
    if(data.loc[observation_index, 'Watched the Star Wars franchise?'] == 'No' or data.loc[observation_index, 'Watched the Star Wars fran
chise?'] == 'Did not answer'):
        if(data.loc[observation_index, 'Fan of the franchise?'] == 'Yes'):
            data.loc[observation_index, 'Fan of the franchise?'] = "No"
    observation_index += 1
observation_index = 2

# Generate all character rating questions
character_rating = []
character_index = 15
while True:
  character_rating.append(data.columns[character_index])
  character_index += 1
  if (character_index == 29):
    break

# Task 1.7: Missing values as well as replace the text value to a readible and sensible data collection
for col in character_rating:
    data[col].replace('Neither favorably nor unfavorably (neutral)','Neutral',inplace=True)
    data[col].replace('Unfamiliar (N/A)','Unfamiliar',inplace=True)
    data[col].fillna('Did not answer', inplace=True)

# Task 1.7: Missing values
data['Which character shot first?'].fillna("Did not answer",inplace=True)
# Now these columns has "No" and "Did not answer" and "Yes"
data['Familiar with the Expanded Universe?'].fillna("Did not answer",inplace=True)
data['Fan of the Expanded Universe?'].fillna("Did not answer",inplace=True)

# Task 1.6: Sanity checks
# If they are a fan of the Expanded Universe, they must be familiar with it
while (observation_index <= max_observation_index):
    if(data.loc[observation_index, 'Fan of the Expanded Universe?'] == 'Yes'):
        if(data.loc[observation_index, 'Familiar with the Expanded Universe?'] == 'No' or data.loc[observation_index, 'Familiar with the
 Expanded Universe?'] == 'Did not answer'):
            data.loc[observation_index, 'Familiar with the Expanded Universe?'] = "Yes"
    observation_index += 1
observation_index = 2

# Task 1.7: Missing values
data['Fan of the Star Trek franchise?'].fillna("Did not answer",inplace=True)
data['Gender'].fillna("Did not answer",inplace=True)

#Task 1.3: Typos (Checking typos for gender) this step is also being repeated everywhere that uses .loc and .replace functions
# First use data['Gender'].value_counts() to see what sort of variations we have
# Then change female => Female, male => Male, F => Male
while (observation_index <= max_observation_index):
    if(data.loc[observation_index, 'Gender'] != 'Male' or data.loc[observation_index, 'Gender'] != 'Female'):
        if(data.loc[observation_index, 'Gender'] == 'female' or data.loc[observation_index, 'Gender'] == 'F'):
            data.loc[observation_index, 'Gender'] = "Female"
        if(data.loc[observation_index, 'Gender'] == 'male' or data.loc[observation_index, 'Gender'] == 'M'):
            data.loc[observation_index, 'Gender'] = "Male"
    observation_index += 1
observation_index = 2

# Task 1.7: Missing values
data['Age'].fillna("Did not answer",inplace=True)
data['Household Income'].fillna("Did not answer",inplace=True)
data['Education'].fillna("Did not answer",inplace=True)
data['Location (Census Region)'].fillna("Did not answer",inplace=True)

#Task 1.6: Sanity checks, checking if answer is reasonable within the age rage seen from .value_counts()
age_response = ['18-29','30-44','45-60','> 60']
while (observation_index <= max_observation_index):
    if(data.loc[observation_index, 'Age'] not in age_response):
        data.loc[observation_index, 'Age'] = "Did not answer"
    observation_index += 1
observation_index = 2

#Task 1.6: Sanity checks, if they have selected a film, they MUST have watched the Star Wars franchise
watched_star_wars = False
while(observation_index <= max_observation_index):
    for col in watchedColumns:
        if(data.loc[observation_index, col] == 'Watched'):
            watched_star_wars = True
    if(watched_star_wars is True):
        data.loc[observation_index, 'Watched the Star Wars franchise?'] = 'Yes'
    else:
        data.loc[observation_index, 'Watched the Star Wars franchise?'] = 'No'
    observation_index += 1
    watched_star_wars = False
observation_index = 2

#Task 1.5: Upper/Lower-case & Task 1.4: Extra-whitespaces (Specifying the column name so it can: UPPERCASE & strip top down)
for header in data.columns:
    data[header] = data[header].str.upper()
    data[header] = data[header].str.strip()

#Task 1.2: Check data types (Ensuring that the appropriate types are being used for future data preparation)
for val in filmRating:
    data[val] = data[val].astype(int)
data['Respondent ID'] = data['Respondent ID'].astype(float)

# Generate all character rating questions
did_not_answer_questions =[]
```

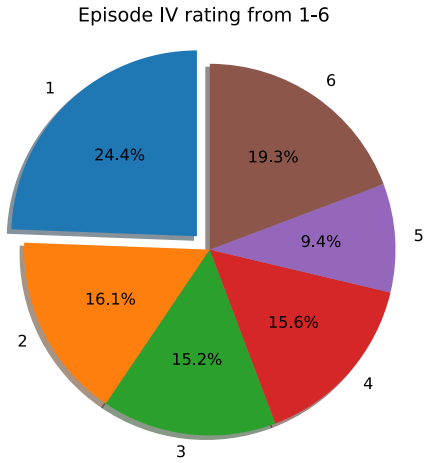
```
question_index = 33
while True:
  did_not_answer_questions.append(data.columns[question_index])
  question_index += 1
  if (question_index == 37):
    break

#Task 1.7: Missing values (Handling observations that did not answer any question at all - omit them)
no_answer = 0
while(observation_index <= max_observation_index):
    if(data.loc[observation_index,'Watched the Star Wars franchise?'] == 'NO' or data.loc[observation_index,'Watched the Star Wars franch
ise?'] == 'DID NOT ANSWER'):
        for val in did_not_answer_questions:
            if(data.loc[observation_index,val] == 'DID NOT ANSWER'):
                no_answer += 1
        if(no_answer == 4):
            data = data.drop(observation_index)
    no_answer = 0
    observation_index += 1
observation_index = 2
```

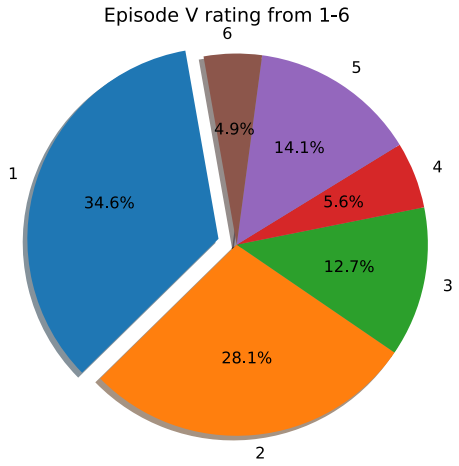
In [2]:
```
# Task 2: Data Exploration
# Task 2.1: Explore a survey question

# Pie chart ordered in anti-clockwise representing how episode I to VI are rated
import matplotlib.pyplot as plt
import matplotlib.pyplot as plt; plt.rcdefaults()

ep = ['Ep I', 'Ep II', 'Ep III', 'Ep IV', 'Ep V', 'Ep VI']
for i in range(len(episodes)):
    episodes[i] = ep[i]

titles = ['Episode I rating from 1-6',
          'Episode II rating from 1-6',
          'Episode III rating from 1-6',
          'Episode IV rating from 1-6',
          'Episode V rating from 1-6',
          'Episode VI rating from 1-6']
labels = ['1','2','3','4','5','6']
explode1 = (0, 0, 0, 0.1, 0, 0)
explode2 = (0, 0, 0, 0, 0.1, 0)
explode3 = (0, 0, 0, 0, 0, 0.1)
explode4 = (0.1, 0, 0, 0, 0, 0)
explode5 = (0.1, 0, 0, 0, 0, 0)
explode6 = (0, 0.1, 0, 0, 0, 0)
episode_counter = 1
title_counter = 0

for episode_number in filmRating:
    sizes = data[episode_number].value_counts().sort_index().tolist()
    sizes.pop(0)
    fig1, ax1 = plt.subplots()
    if(episode_counter == 1):
        ax1.pie(sizes, explode=explode1, labels=labels, autopct='%1.1f%%', shadow=True, startangle=90)
    if(episode_counter == 2):
        ax1.pie(sizes, explode=explode2, labels=labels, autopct='%1.1f%%', shadow=True, startangle=90)
    if(episode_counter == 3):
        ax1.pie(sizes, explode=explode3, labels=labels, autopct='%1.1f%%', shadow=True, startangle=90)
    if(episode_counter == 4):
        ax1.pie(sizes, explode=explode4, labels=labels, autopct='%1.1f%%', shadow=True, startangle=90)
    if(episode_counter == 5):
        ax1.pie(sizes, explode=explode5, labels=labels, autopct='%1.1f%%', shadow=True, startangle=100)
    if(episode_counter == 6):
        ax1.pie(sizes, explode=explode6, labels=labels, autopct='%1.1f%%', shadow=True, startangle=90)
    ax1.axis('equal')  # Equal aspect ratio ensures that pie is drawn as a circle.
    episode_counter += 1
    plt.title(titles[title_counter])
    title_counter += 1
    plt.show()

import numpy as np

# Gather the film rating from plots from previous step
rating_percentage = [17.5,34.6,24.4,4.3,3.8,15.4]
titles = ('Return of The Jedi', 'The Empire Strikes Back', 'A New Hope', 'Revenge of The Sith','Attack of The Clones','The Phantom Menac
e')
fig, ax = plt.subplots()
for i, v in enumerate(rating_percentage):
    ax.text(v + 0.5, i, str(v), color='k', fontweight='bold')
for spine in ax.spines:
    ax.spines[spine].set_visible(False)

y_pos = np.arange(len(titles))
plt.barh(y_pos, rating_percentage)
ax.xaxis.set_visible(False)
plt.yticks(y_pos, titles)

plt.title('How many Star Wars audience rate each film to be the best of the franchise in %')
plt.show()
```

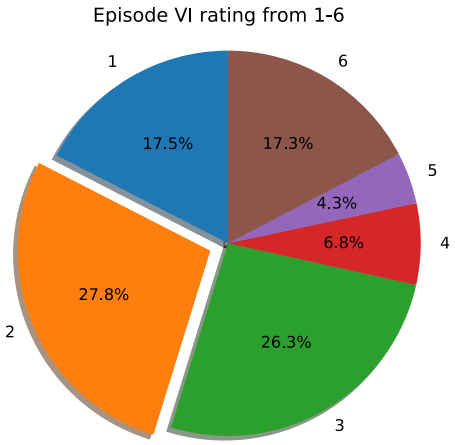

Episode I rating from 1-6
1
6
15.4%
20.1%
2
8.5%
12.0%
5
15.6%
3
28.4%
4

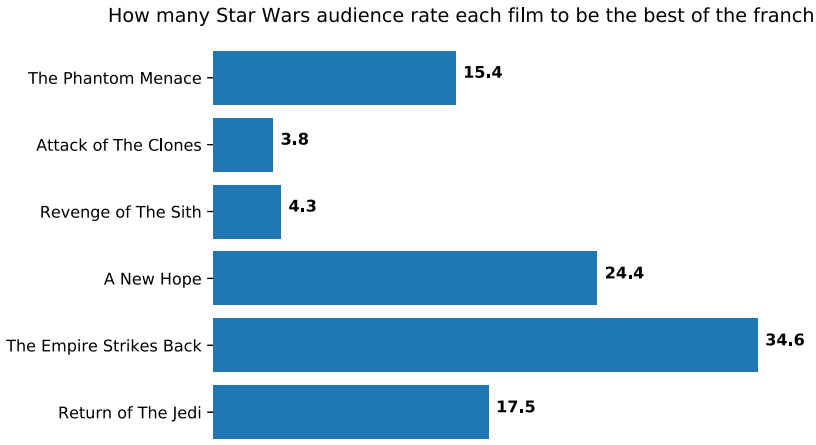

Episode II rating from 1-6
1
6
2
3.8%
12.2%
13.9%
3
12.3%
35.9%
5
21.9%
4


Episode III rating from 1-6
1
2
6
4.3%
5.6%
26.0%
3
18.0%
21.8%
24.3%
4
5

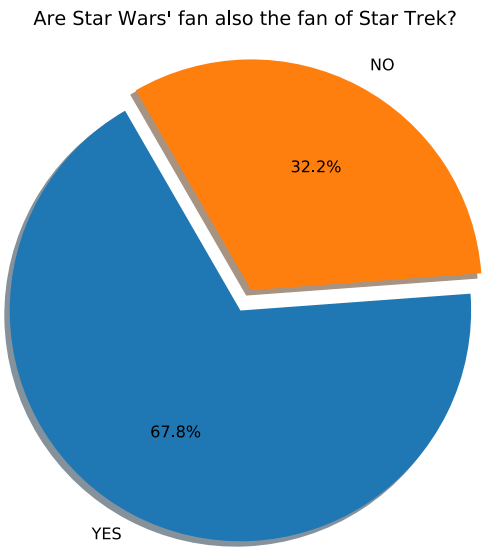

Episode IV rating from 1-6
1
6
24.4%
19.3%
9.4%
5
16.1%
2
15.6%
15.2%
4
3


Episode V rating from 1-6
6
5
4
3
2
1
4.9%
14.1%
5.6%
12.7%
28.1%
34.6%

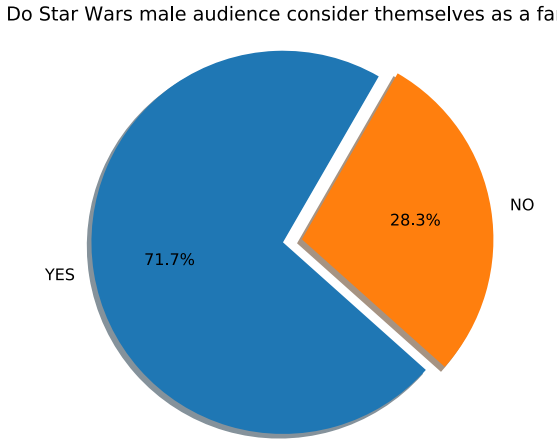

Episode VI rating from 1-6
1
6
5
4
3
2
17.5%
17.3%
4.3%
6.8%
26.3%
27.8%


How many Star Wars audience rate each film to be the best of the franchise in %
The Phantom Menace
Attack of The Clones
Revenge of The Sith
A New Hope
The Empire Strikes Back
Return of The Jedi
15.4
3.8
4.3
24.4
34.6
17.5

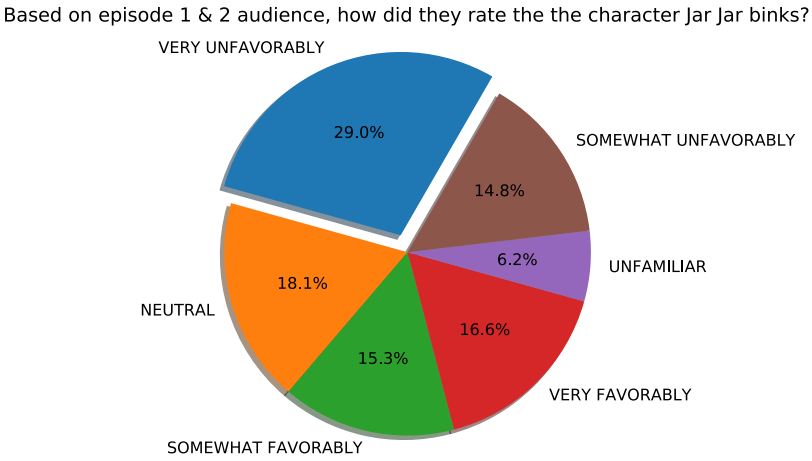
In [11]:
```
# Task 2.2:
# Hypothesis 1: People who is a fan of the Star Wars Franchise, also is a fan of the Star Trek franchise

# Analyse all "Fan of the franchise?" = 'YES' with "Fan of the Star Trek franchise?" = 'YES' and 'NO'
star_trek_fan = []
observation_index = 2
while(observation_index <= max_observation_index):
    if(observation_index in data.index):
        if(data.loc[observation_index,'Fan of the franchise?'] == 'YES'):
            star_trek_fan.append(data.loc[observation_index,'Fan of the Star Trek franchise?'])
    observation_index += 1
observation_index = 2

df = []
df.append(star_trek_fan.count('YES'))
df.append(star_trek_fan.count('NO'))
# df.append(star_trek_fan.count('DID NOT ANSWER'))

fan_base_title = ['YES','NO']

explode = (0.1, 0)

fig1, ax1 = plt.subplots()
ax1.pie(df, explode=explode, labels=fan_base_title, autopct='%1.1f%%',
        shadow=True, startangle=120)
ax1.axis('equal')  # Equal aspect ratio ensures that pie is drawn as a circle.
plt.title("Are Star Wars' fan also the fan of Star Trek?")
plt.show()
```



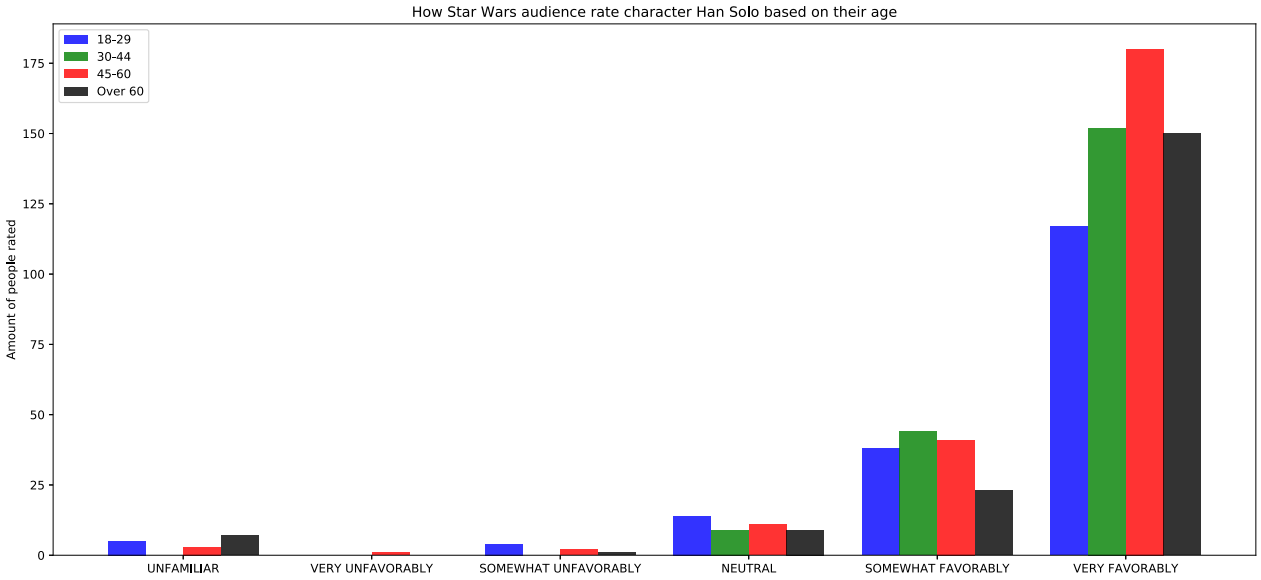
In [4]:
```
# Task 2.2:
# Hypothesis 2: Majority of men who watch Star Wars consider themselves as a fan

# Analyse all "Gender" = 'MALE' with "Fan of the franchise?" = 'YES' and 'NO'
male_audience = []
observation_index = 2
while(observation_index <= max_observation_index):
    if(observation_index in data.index):
        if(data.loc[observation_index,'Gender'] == "MALE"):
            male_audience.append(data.loc[observation_index,'Fan of the franchise?'])
    observation_index += 1
observation_index = 2

male_audience_count = []
male_audience_count.append(male_audience.count('YES'))
male_audience_count.append(male_audience.count('NO'))

response_labels = ['YES','NO']

explode = (0.1, 0)

fig1, ax1 = plt.subplots()
ax1.pie(male_audience_count, explode=explode, labels=response_labels, autopct='%1.1f%%',
        shadow=True, startangle=60)
ax1.axis('equal')
plt.title("Do Star Wars male audience consider themselves as a fan?")
plt.show()
```



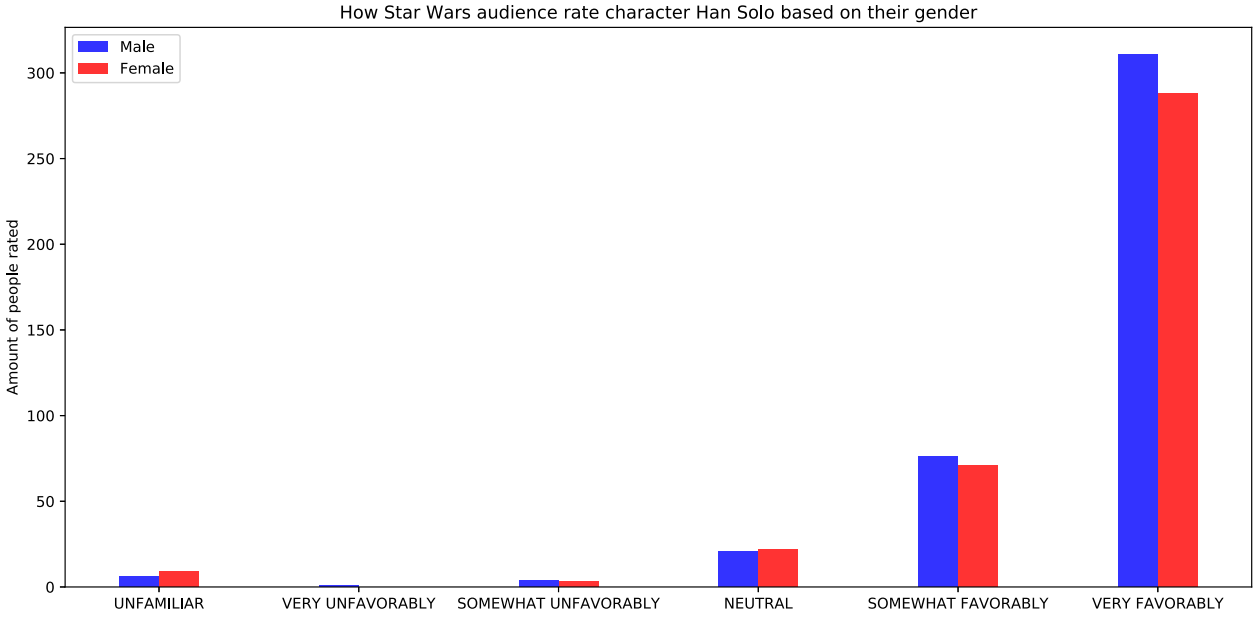
In [5]:
```
# Task 2.2:
# Hypothesis 3: Majority of the audience who rate episode 2 as the favorite film, also rate the character Jar Jar Binks to be very favora
bly

# Analyse all "Gender" = 'MALE' with "Fan of the franchise?" = 'YES' and 'NO'
audience = []
observation_index = 2
while(observation_index <= max_observation_index):
    if(observation_index in data.index):
        if(data.loc[observation_index,'Seen Ep I'] == "WATCHED" and data.loc[observation_index,'Seen Ep II'] == "WATCHED"):
            audience.append(data.loc[observation_index,'How do you like the character: Jar Jar Binks'])
    observation_index += 1
observation_index = 2

audience_count = []
audience_count.append(audience.count('VERY UNFAVORABLY'))
audience_count.append(audience.count('NEUTRAL'))
audience_count.append(audience.count('SOMEWHAT FAVORABLY'))
audience_count.append(audience.count('VERY FAVORABLY'))
audience_count.append(audience.count('UNFAMILIAR'))
audience_count.append(audience.count('SOMEWHAT UNFAVORABLY'))

response_labels = ['VERY UNFAVORABLY','NEUTRAL','SOMEWHAT FAVORABLY','VERY FAVORABLY','UNFAMILIAR','SOMEWHAT UNFAVORABLY']

explode = (0.1, 0, 0, 0, 0, 0)

fig1, ax1 = plt.subplots()
ax1.pie(audience_count, explode=explode, labels=response_labels, autopct='%1.1f%%',
        shadow=True, startangle=60)
ax1.axis('equal')
plt.title("Based on episode 1 & 2 audience, how did they rate the the character Jar Jar binks?")
plt.show()
```



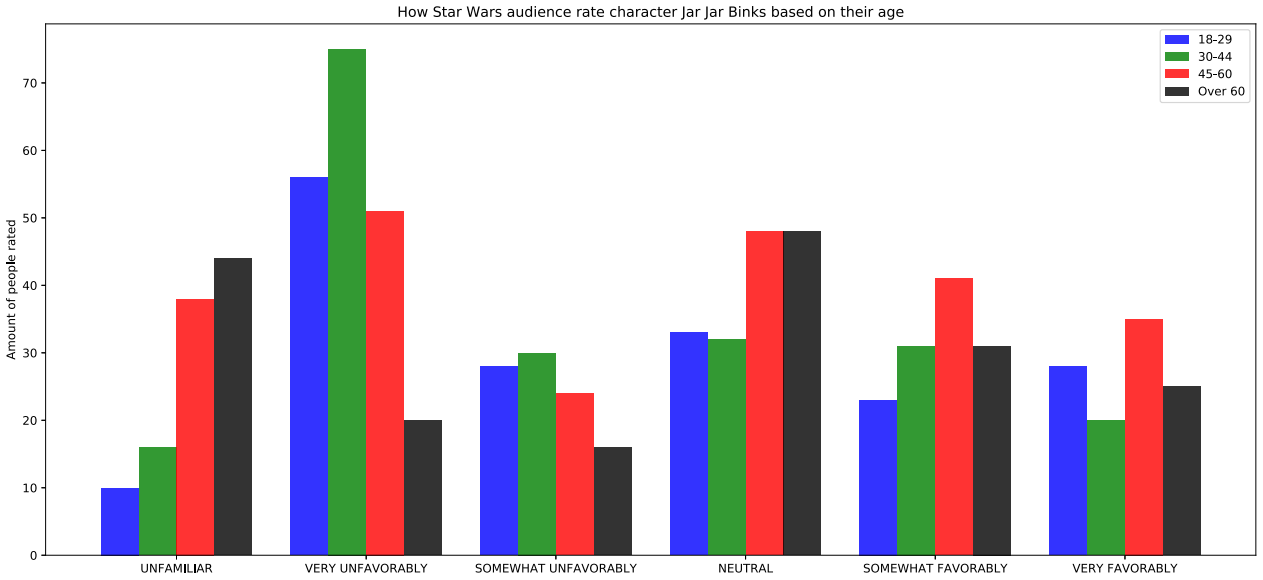
In [6]:
```
# Task 2.3
# Following code will generate a bar chart of how people rate the character Han Solo based on their age

labels = ['UNFAMILIAR', 'VERY UNFAVORABLY', 'SOMEWHAT UNFAVORABLY', 'NEUTRAL', 'SOMEWHAT FAVORABLY', 'VERY FAVORABLY']
han_solo_rating18to29 = []
han_solo_rating18to29_count = []
han_solo_rating30to44 = []
han_solo_rating30to44_count = []
han_solo_rating45to60 = []
han_solo_rating45to60_count = []
han_solo_ratingover60 = []
han_solo_ratingover60_count = []

observation_index = 2
while(observation_index <= max_observation_index):
    if(observation_index in data.index):
        if(data.loc[observation_index,'Age'] == "18-29" and data.loc[observation_index,'How do you like the character: Han Solo'] != "DID
NOT ANSWER"):
            han_solo_rating18to29.append(data.loc[observation_index,'How do you like the character: Han Solo'])
        if(data.loc[observation_index,'Age'] == "30-44" and data.loc[observation_index,'How do you like the character: Han Solo'] != "DID
NOT ANSWER"):
            han_solo_rating30to44.append(data.loc[observation_index,'How do you like the character: Han Solo'])
        if(data.loc[observation_index,'Age'] == "45-60" and data.loc[observation_index,'How do you like the character: Han Solo'] != "DID
NOT ANSWER"):
            han_solo_rating45to60.append(data.loc[observation_index,'How do you like the character: Han Solo'])
        if(data.loc[observation_index,'Age'] == "> 60" and data.loc[observation_index,'How do you like the character: Han Solo'] != "DID
 NOT ANSWER"):
            han_solo_ratingover60.append(data.loc[observation_index,'How do you like the character: Han Solo'])
    observation_index += 1
observation_index = 2

han_solo_rating18to29_count.append(han_solo_rating18to29.count('UNFAMILIAR'))
han_solo_rating18to29_count.append(han_solo_rating18to29.count('VERY UNFAVORABLY'))
han_solo_rating18to29_count.append(han_solo_rating18to29.count('SOMEWHAT UNFAVORABLY'))
han_solo_rating18to29_count.append(han_solo_rating18to29.count('NEUTRAL'))
han_solo_rating18to29_count.append(han_solo_rating18to29.count('SOMEWHAT FAVORABLY'))
han_solo_rating18to29_count.append(han_solo_rating18to29.count('VERY FAVORABLY'))

han_solo_rating30to44_count.append(han_solo_rating30to44.count('UNFAMILIAR'))
han_solo_rating30to44_count.append(han_solo_rating30to44.count('VERY UNFAVORABLY'))
han_solo_rating30to44_count.append(han_solo_rating30to44.count('SOMEWHAT UNFAVORABLY'))
han_solo_rating30to44_count.append(han_solo_rating30to44.count('NEUTRAL'))
han_solo_rating30to44_count.append(han_solo_rating30to44.count('SOMEWHAT FAVORABLY'))
han_solo_rating30to44_count.append(han_solo_rating30to44.count('VERY FAVORABLY'))

han_solo_rating45to60_count.append(han_solo_rating45to60.count('UNFAMILIAR'))
han_solo_rating45to60_count.append(han_solo_rating45to60.count('VERY UNFAVORABLY'))
han_solo_rating45to60_count.append(han_solo_rating45to60.count('SOMEWHAT UNFAVORABLY'))
han_solo_rating45to60_count.append(han_solo_rating45to60.count('NEUTRAL'))
han_solo_rating45to60_count.append(han_solo_rating45to60.count('SOMEWHAT FAVORABLY'))
han_solo_rating45to60_count.append(han_solo_rating45to60.count('VERY FAVORABLY'))

han_solo_ratingover60_count.append(han_solo_ratingover60.count('UNFAMILIAR'))
han_solo_ratingover60_count.append(han_solo_ratingover60.count('VERY UNFAVORABLY'))
han_solo_ratingover60_count.append(han_solo_ratingover60.count('SOMEWHAT UNFAVORABLY'))
han_solo_ratingover60_count.append(han_solo_ratingover60.count('NEUTRAL'))
han_solo_ratingover60_count.append(han_solo_ratingover60.count('SOMEWHAT FAVORABLY'))
han_solo_ratingover60_count.append(han_solo_ratingover60.count('VERY FAVORABLY'))

n_groups = 6

# create plot
plt.rcParams['figure.figsize'] = [15, 7]
index = np.arange(n_groups)
bar_width = 0.2
opacity = 0.8

rects1 = plt.bar(index, han_solo_rating18to29_count, bar_width,alpha=opacity,color='b',label='18-29')
rects2 = plt.bar(index + bar_width, han_solo_rating30to44_count, bar_width,alpha=opacity,color='g',label='30-44')
rects3 = plt.bar(index + bar_width + bar_width, han_solo_rating45to60_count, bar_width,alpha=opacity,color='r',label='45-60')
rects4 = plt.bar(index + bar_width + bar_width + bar_width, han_solo_ratingover60_count, bar_width,alpha=opacity,color='k',label='Over 6
0')

plt.ylabel('Amount of people rated')
plt.title('How Star Wars audience rate character Han Solo based on their age')
plt.xticks(index + bar_width + 0.1, labels)
plt.legend()

plt.tight_layout()
plt.show()
```


How Star Wars audience rate character Han Solo based on their age
18-29
30-44
45-60
Over 60
Amount of people rated
0
25
50
75
100
125
150
175
UNFAMILIAR
VERY UNFAVORABLY
SOMEWHAT UNFAVORABLY
NEUTRAL
SOMEWHAT FAVORABLY
VERY FAVORABLY

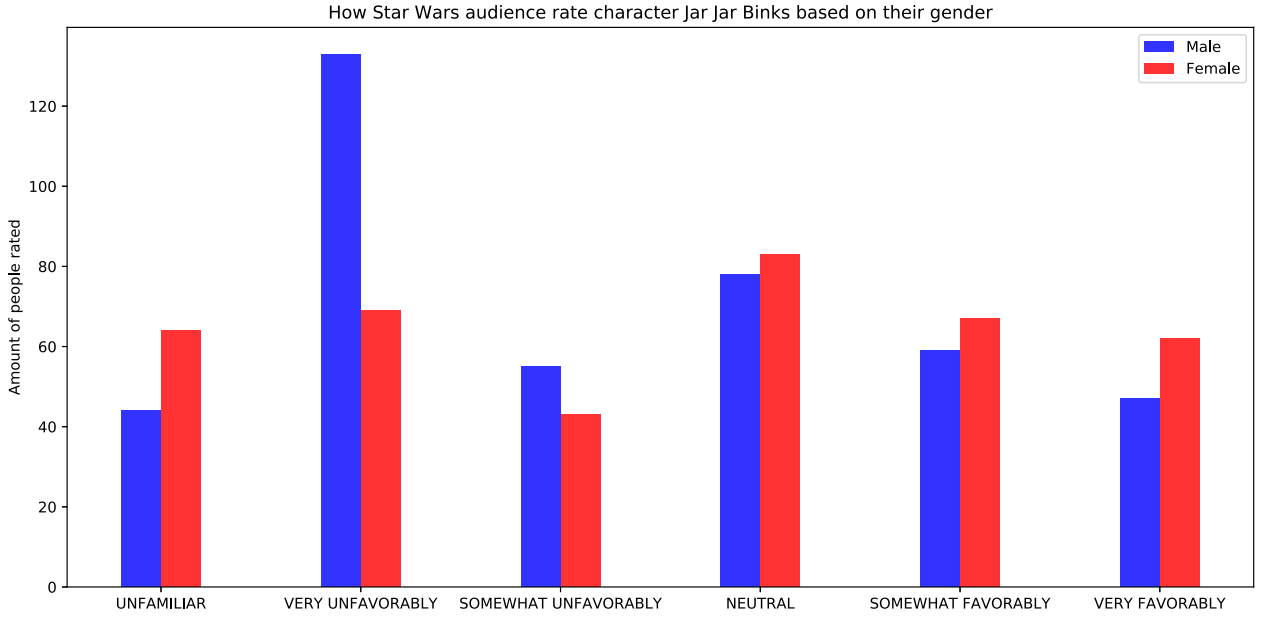
In [7]:
```
# Task 2.3
# Following code will generate a bar chart of how people rate the character Han Solo based on their gender

labels = ['UNFAMILIAR', 'VERY UNFAVORABLY', 'SOMEWHAT UNFAVORABLY', 'NEUTRAL', 'SOMEWHAT FAVORABLY', 'VERY FAVORABLY']
male_rating = []
male_rating_count = []
female_rating = []
female_rating_count = []


observation_index = 2
while(observation_index <= max_observation_index):
    if(observation_index in data.index):
        if(data.loc[observation_index,'Gender'] == "MALE" and data.loc[observation_index,'How do you like the character: Han Solo'] != "D
ID NOT ANSWER"):
            male_rating.append(data.loc[observation_index,'How do you like the character: Han Solo'])
        if(data.loc[observation_index,'Gender'] == "FEMALE" and data.loc[observation_index,'How do you like the character: Han Solo'] !=
"DID NOT ANSWER"):
            female_rating.append(data.loc[observation_index,'How do you like the character: Han Solo'])
    observation_index += 1
observation_index = 2

male_rating_count.append(male_rating.count('UNFAMILIAR'))
male_rating_count.append(male_rating.count('VERY UNFAVORABLY'))
male_rating_count.append(male_rating.count('SOMEWHAT UNFAVORABLY'))
male_rating_count.append(male_rating.count('NEUTRAL'))
male_rating_count.append(male_rating.count('SOMEWHAT FAVORABLY'))
male_rating_count.append(male_rating.count('VERY FAVORABLY'))

female_rating_count.append(female_rating.count('UNFAMILIAR'))
female_rating_count.append(female_rating.count('VERY UNFAVORABLY'))
female_rating_count.append(female_rating.count('SOMEWHAT UNFAVORABLY'))
female_rating_count.append(female_rating.count('NEUTRAL'))
female_rating_count.append(female_rating.count('SOMEWHAT FAVORABLY'))
female_rating_count.append(female_rating.count('VERY FAVORABLY'))

n_groups = 6

# create plot
plt.rcParams['figure.figsize'] = [12, 6]
index = np.arange(n_groups)
bar_width = 0.2
opacity = 0.8

rects1 = plt.bar(index, male_rating_count, bar_width,alpha=opacity,color='b',label='Male')
rects2 = plt.bar(index + bar_width, female_rating_count, bar_width,alpha=opacity,color='r',label='Female')

plt.ylabel('Amount of people rated')
plt.title('How Star Wars audience rate character Han Solo based on their gender')
plt.xticks(index + 0.1, labels)
plt.legend()

plt.tight_layout()
plt.show()
```



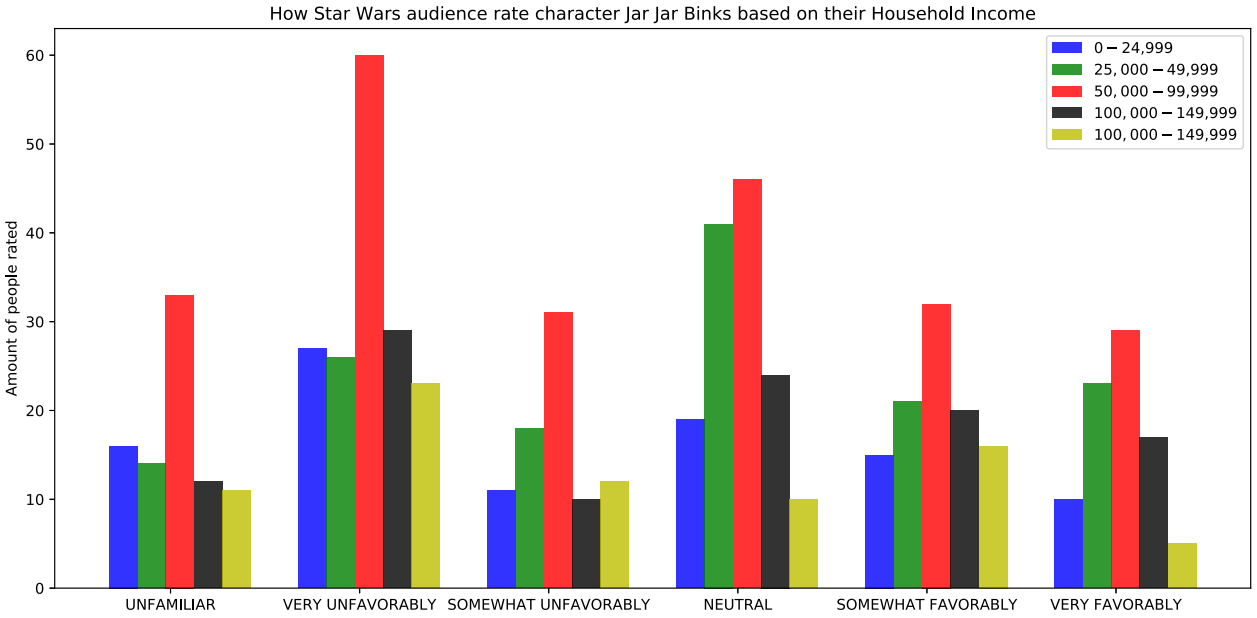
In [8]:
```
# Task 2.3
# Following code will generate a bar chart of how people rate the character Jar Jar Binks based on their age

jarjar_rating18to29 = []
jarjar_rating18to29_count = []
jarjar_rating30to44 = []
jarjar_rating30to44_count = []
jarjar_rating45to60 = []
jarjar_rating45to60_count = []
jarjar_ratingover60 = []
jarjar_ratingover60_count = []

observation_index = 2
while(observation_index <= max_observation_index):
    if(observation_index in data.index):
        if(data.loc[observation_index,'Age'] == "18-29" and data.loc[observation_index,'How do you like the character: Jar Jar Binks'] !=
"DID NOT ANSWER"):
            jarjar_rating18to29.append(data.loc[observation_index,'How do you like the character: Jar Jar Binks'])
        if(data.loc[observation_index,'Age'] == "30-44" and data.loc[observation_index,'How do you like the character: Jar Jar Binks'] !=
"DID NOT ANSWER"):
            jarjar_rating30to44.append(data.loc[observation_index,'How do you like the character: Jar Jar Binks'])
        if(data.loc[observation_index,'Age'] == "45-60" and data.loc[observation_index,'How do you like the character: Jar Jar Binks'] !=
"DID NOT ANSWER"):
            jarjar_rating45to60.append(data.loc[observation_index,'How do you like the character: Jar Jar Binks'])
        if(data.loc[observation_index,'Age'] == "> 60" and data.loc[observation_index,'How do you like the character: Jar Jar Binks'] !=
"DID NOT ANSWER"):
            jarjar_ratingover60.append(data.loc[observation_index,'How do you like the character: Jar Jar Binks'])
    observation_index += 1
observation_index = 2

jarjar_rating18to29_count.append(jarjar_rating18to29.count('UNFAMILIAR'))
jarjar_rating18to29_count.append(jarjar_rating18to29.count('VERY UNFAVORABLY'))
jarjar_rating18to29_count.append(jarjar_rating18to29.count('SOMEWHAT UNFAVORABLY'))
jarjar_rating18to29_count.append(jarjar_rating18to29.count('NEUTRAL'))
jarjar_rating18to29_count.append(jarjar_rating18to29.count('SOMEWHAT FAVORABLY'))
jarjar_rating18to29_count.append(jarjar_rating18to29.count('VERY FAVORABLY'))

jarjar_rating30to44_count.append(jarjar_rating30to44.count('UNFAMILIAR'))
jarjar_rating30to44_count.append(jarjar_rating30to44.count('VERY UNFAVORABLY'))
jarjar_rating30to44_count.append(jarjar_rating30to44.count('SOMEWHAT UNFAVORABLY'))
jarjar_rating30to44_count.append(jarjar_rating30to44.count('NEUTRAL'))
jarjar_rating30to44_count.append(jarjar_rating30to44.count('SOMEWHAT FAVORABLY'))
jarjar_rating30to44_count.append(jarjar_rating30to44.count('VERY FAVORABLY'))

jarjar_rating45to60_count.append(jarjar_rating45to60.count('UNFAMILIAR'))
jarjar_rating45to60_count.append(jarjar_rating45to60.count('VERY UNFAVORABLY'))
jarjar_rating45to60_count.append(jarjar_rating45to60.count('SOMEWHAT UNFAVORABLY'))
jarjar_rating45to60_count.append(jarjar_rating45to60.count('NEUTRAL'))
jarjar_rating45to60_count.append(jarjar_rating45to60.count('SOMEWHAT FAVORABLY'))
jarjar_rating45to60_count.append(jarjar_rating45to60.count('VERY FAVORABLY'))

jarjar_ratingover60_count.append(jarjar_ratingover60.count('UNFAMILIAR'))
jarjar_ratingover60_count.append(jarjar_ratingover60.count('VERY UNFAVORABLY'))
jarjar_ratingover60_count.append(jarjar_ratingover60.count('SOMEWHAT UNFAVORABLY'))
jarjar_ratingover60_count.append(jarjar_ratingover60.count('NEUTRAL'))
jarjar_ratingover60_count.append(jarjar_ratingover60.count('SOMEWHAT FAVORABLY'))
jarjar_ratingover60_count.append(jarjar_ratingover60.count('VERY FAVORABLY'))

n_groups = 6

# create plot
plt.rcParams['figure.figsize'] = [15, 7]
index = np.arange(n_groups)
bar_width = 0.2
opacity = 0.8

rects1 = plt.bar(index, jarjar_rating18to29_count, bar_width,alpha=opacity,color='b',label='18-29')
rects2 = plt.bar(index + bar_width, jarjar_rating30to44_count, bar_width,alpha=opacity,color='g',label='30-44')
rects3 = plt.bar(index + bar_width + bar_width, jarjar_rating45to60_count, bar_width,alpha=opacity,color='r',label='45-60')
rects4 = plt.bar(index + bar_width + bar_width + bar_width, jarjar_ratingover60_count, bar_width,alpha=opacity,color='k',label='Over 60')

plt.ylabel('Amount of people rated')
plt.title('How Star Wars audience rate character Jar Jar Binks based on their age')
plt.xticks(index + bar_width + 0.1, labels)
plt.legend()

plt.tight_layout()
plt.show()
```


How Star Wars audience rate character Jar Jar Binks based on their age
Amount of people rated
0
10
20
30
40
50
60
70
18-29
30-44
45-60
Over 60
UNFAMILIAR
VERY UNFAVORABLY
SOMEWHAT UNFAVORABLY
NEUTRAL
SOMEWHAT FAVORABLY
VERY FAVORABLY

In [9]:
```
# Task 2.3
# Following code will generate a bar chart of how people rate the character Jar Jar Binks based on their gender

labels = ['UNFAMILIAR', 'VERY UNFAVORABLY', 'SOMEWHAT UNFAVORABLY', 'NEUTRAL', 'SOMEWHAT FAVORABLY', 'VERY FAVORABLY']
male_rating = []
male_rating_count = []
female_rating = []
female_rating_count = []


observation_index = 2
while(observation_index <= max_observation_index):
    if(observation_index in data.index):
        if(data.loc[observation_index,'Gender'] == "MALE" and data.loc[observation_index,'How do you like the character: Jar Jar Binks']
!= "DID NOT ANSWER"):
            male_rating.append(data.loc[observation_index,'How do you like the character: Jar Jar Binks'])
        if(data.loc[observation_index,'Gender'] == "FEMALE" and data.loc[observation_index,'How do you like the character: Jar Jar Binks'
] != "DID NOT ANSWER"):
            female_rating.append(data.loc[observation_index,'How do you like the character: Jar Jar Binks'])
    observation_index += 1
observation_index = 2

male_rating_count.append(male_rating.count('UNFAMILIAR'))
male_rating_count.append(male_rating.count('VERY UNFAVORABLY'))
male_rating_count.append(male_rating.count('SOMEWHAT UNFAVORABLY'))
male_rating_count.append(male_rating.count('NEUTRAL'))
male_rating_count.append(male_rating.count('SOMEWHAT FAVORABLY'))
male_rating_count.append(male_rating.count('VERY FAVORABLY'))

female_rating_count.append(female_rating.count('UNFAMILIAR'))
female_rating_count.append(female_rating.count('VERY UNFAVORABLY'))
female_rating_count.append(female_rating.count('SOMEWHAT UNFAVORABLY'))
female_rating_count.append(female_rating.count('NEUTRAL'))
female_rating_count.append(female_rating.count('SOMEWHAT FAVORABLY'))
female_rating_count.append(female_rating.count('VERY FAVORABLY'))

n_groups = 6

# create plot
plt.rcParams['figure.figsize'] = [12, 6]
index = np.arange(n_groups)
bar_width = 0.2
opacity = 0.8

rects1 = plt.bar(index, male_rating_count, bar_width,alpha=opacity,color='b',label='Male')
rects2 = plt.bar(index + bar_width, female_rating_count, bar_width,alpha=opacity,color='r',label='Female')

plt.ylabel('Amount of people rated')
plt.title('How Star Wars audience rate character Jar Jar Binks based on their gender')
plt.xticks(index + 0.1, labels)
plt.legend()

plt.tight_layout()
plt.show()
```



In [10]:
```
# Task 2.3
# Following code will generate a bar chart of how people rate the character Jar Jar Binks based on their household income

jarjar_rating0to24 = []
jarjar_rating0to24_count = []
jarjar_rating25to49 = []
jarjar_rating25to49_count = []
jarjar_rating50to99 = []
jarjar_rating50to99_count = []
jarjar_rating100to149 = []
jarjar_rating100to149_count = []
jarjar_ratingover150 = []
jarjar_ratingover150_count = []

observation_index = 2
while(observation_index <= max_observation_index):
    if(observation_index in data.index):
        if(data.loc[observation_index,'Household Income'] == "$0 - $24,999" and data.loc[observation_index,'How do you like the characte
r: Jar Jar Binks'] != "DID NOT ANSWER"):
            jarjar_rating0to24.append(data.loc[observation_index,'How do you like the character: Jar Jar Binks'])
        if(data.loc[observation_index,'Household Income'] == "$25,000 - $49,999" and data.loc[observation_index,'How do you like the char
acter: Jar Jar Binks'] != "DID NOT ANSWER"):
            jarjar_rating25to49.append(data.loc[observation_index,'How do you like the character: Jar Jar Binks'])
        if(data.loc[observation_index,'Household Income'] == "$50,000 - $99,999" and data.loc[observation_index,'How do you like the char
acter: Jar Jar Binks'] != "DID NOT ANSWER"):
            jarjar_rating50to99.append(data.loc[observation_index,'How do you like the character: Jar Jar Binks'])
        if(data.loc[observation_index,'Household Income'] == "$100,000 - $149,999" and data.loc[observation_index,'How do you like the ch
aracter: Jar Jar Binks'] != "DID NOT ANSWER"):
            jarjar_rating100to149.append(data.loc[observation_index,'How do you like the character: Jar Jar Binks'])
        if(data.loc[observation_index,'Household Income'] == "$150,000+" and data.loc[observation_index,'How do you like the character: J
ar Jar Binks'] != "DID NOT ANSWER"):
            jarjar_ratingover150.append(data.loc[observation_index,'How do you like the character: Jar Jar Binks'])
    observation_index += 1
observation_index = 2

jarjar_rating0to24_count.append(jarjar_rating0to24.count('UNFAMILIAR'))
jarjar_rating0to24_count.append(jarjar_rating0to24.count('VERY UNFAVORABLY'))
jarjar_rating0to24_count.append(jarjar_rating0to24.count('SOMEWHAT UNFAVORABLY'))
jarjar_rating0to24_count.append(jarjar_rating0to24.count('NEUTRAL'))
jarjar_rating0to24_count.append(jarjar_rating0to24.count('SOMEWHAT FAVORABLY'))
jarjar_rating0to24_count.append(jarjar_rating0to24.count('VERY FAVORABLY'))

jarjar_rating25to49_count.append(jarjar_rating25to49.count('UNFAMILIAR'))
jarjar_rating25to49_count.append(jarjar_rating25to49.count('VERY UNFAVORABLY'))
jarjar_rating25to49_count.append(jarjar_rating25to49.count('SOMEWHAT UNFAVORABLY'))
jarjar_rating25to49_count.append(jarjar_rating25to49.count('NEUTRAL'))
jarjar_rating25to49_count.append(jarjar_rating25to49.count('SOMEWHAT FAVORABLY'))
jarjar_rating25to49_count.append(jarjar_rating25to49.count('VERY FAVORABLY'))

jarjar_rating50to99_count.append(jarjar_rating50to99.count('UNFAMILIAR'))
jarjar_rating50to99_count.append(jarjar_rating50to99.count('VERY UNFAVORABLY'))
jarjar_rating50to99_count.append(jarjar_rating50to99.count('SOMEWHAT UNFAVORABLY'))
jarjar_rating50to99_count.append(jarjar_rating50to99.count('NEUTRAL'))
jarjar_rating50to99_count.append(jarjar_rating50to99.count('SOMEWHAT FAVORABLY'))
jarjar_rating50to99_count.append(jarjar_rating50to99.count('VERY FAVORABLY'))

jarjar_rating100to149_count.append(jarjar_rating100to149.count('UNFAMILIAR'))
jarjar_rating100to149_count.append(jarjar_rating100to149.count('VERY UNFAVORABLY'))
jarjar_rating100to149_count.append(jarjar_rating100to149.count('SOMEWHAT UNFAVORABLY'))
jarjar_rating100to149_count.append(jarjar_rating100to149.count('NEUTRAL'))
jarjar_rating100to149_count.append(jarjar_rating100to149.count('SOMEWHAT FAVORABLY'))
jarjar_rating100to149_count.append(jarjar_rating100to149.count('VERY FAVORABLY'))

jarjar_ratingover150_count.append(jarjar_ratingover150.count('UNFAMILIAR'))
jarjar_ratingover150_count.append(jarjar_ratingover150.count('VERY UNFAVORABLY'))
jarjar_ratingover150_count.append(jarjar_ratingover150.count('SOMEWHAT UNFAVORABLY'))
jarjar_ratingover150_count.append(jarjar_ratingover150.count('NEUTRAL'))
jarjar_ratingover150_count.append(jarjar_ratingover150.count('SOMEWHAT FAVORABLY'))
jarjar_ratingover150_count.append(jarjar_ratingover150.count('VERY FAVORABLY'))

n_groups = 6

# create plot
plt.rcParams['figure.figsize'] = [12, 6]
index = np.arange(n_groups)
bar_width = 0.15
opacity = 0.8

rects1 = plt.bar(index, jarjar_rating0to24_count, bar_width,alpha=opacity,color='b',label='$0 - $24,999')
rects2 = plt.bar(index + bar_width, jarjar_rating25to49_count, bar_width,alpha=opacity,color='g',label='$25,000 - $49,999')
rects3 = plt.bar(index + bar_width + bar_width, jarjar_rating50to99_count, bar_width,alpha=opacity,color='r',label='$50,000 - $99,999')
rects4 = plt.bar(index + bar_width + bar_width + bar_width, jarjar_rating100to149_count, bar_width,alpha=opacity,color='k',label='$100,00
0 - $149,999')
rects5 = plt.bar(index + bar_width + bar_width + bar_width + bar_width, jarjar_ratingover150_count, bar_width,alpha=opacity,color='y',lab
el='$100,000 - $149,999')

plt.ylabel('Amount of people rated')
plt.title('How Star Wars audience rate character Jar Jar Binks based on their Household Income')
plt.xticks(index + bar_width + 0.1, labels)
plt.legend()

plt.tight_layout()
plt.show()
```


How Star Wars audience rate character Jar Jar Binks based on their Household Income
Amount of people rated
0
10
20
30
40
50
60
UNFAMILIAR
VERY UNFAVORABLY
SOMEWHAT UNFAVORABLY
NEUTRAL
SOMEWHAT FAVORABLY
VERY FAVORABLY
0 – 24,999
25,000 – 49,999
50,000 – 99,999
100,000 – 149,999
100,000 – 149,999