Attribute Information:

- Income --> customer's yearly household income
- Kidhome --> number of small children in customer's househokl
- Teenhome --> number of teenagers in customer's household
- Recency --> number of days since the last purchase
- MntWines --> amount spent on wines in the last 2 years
- MntFruits --> amount spent on fruits in the last 2 years
- MntMeatProducts --> amount spent on meat products in the last 2 years
- MntFishProducts --> amount spent on fish products in the last 2 years
- MntSweatProducts --> amount spent on sweat products in ths last 2 years
- MntGoldProds --> amount spent on gold products in the last 2 years
- NumDealsPruchases --> number of purchases made with discount
- NumWebPurchases --> number of purchases made through company's web site
- NumCatalogPurchases --> number of purchases made using catalogue
- NumStorePurchases --> number of purchases made directly in stores
- NumWebVsitsMonth --> number of visits to company's web site in the last month
- AcceptedCm3 --> 1 if customer accepted the offer in the 3nd campaign, 0 otherwise
- AcceptedCm4--> 1 if customer accepted the offer in the 5th campaign, 0 otherwise
- AcceptedCm5--> 1 if customer accepted the offer in the 3nd campaign, 0 otherwise
- AcceptedCm1 --> 1 if customer accepted the offer in the 1st campaign, 0 otherwise
- AcceptedCm2 --> 1 if customer accepted the offer in the 2nd campaign, 0 otherwise
- Complain --> 1 if customer complained in the last 2 years
- Z_CostContact
- Z_Revenue
- Response(target) --> 1 if costumer accepted the offer in the last campaign, 0 otherwise
- Age
- Customer_Days
- marital_Divorced
- marital_Married
- marital_Single
- marital_Together
- marital_Widow
- education_2n Cycle
- education_Basic
- education_Graduation
- education_Master
- education_PhD
- MntTotal
- MntRegularProds
- AcceptedCmpOverall

Import Libraries

In [2]:

```
# Import necessary libraries for data analysis and visualization
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
```

Read data

In [3]:

```
# Read in data from a CSV file and store it in a pandas DataFrame
data = pd.read_csv(r"C:\Users\Anups\Desktop\python project\marketing_data.csv")
```

In [4]:

```
# Retrieve the column names of the pandas DataFrame
data.columns
```

Out[4]:

```
Index(['Income', 'Kidhome', 'Teenhome', 'Recency', 'MntWines', 'MntFruits',
       'MntMeatProducts', 'MntFishProducts', 'MntSweetProducts',
       'MntGoldProds', 'NumDealsPurchases', 'NumWebPurchases',
       'NumCatalogPurchases', 'NumStorePurchases', 'NumWebVisitsMonth',
       'AcceptedCmp3', 'AcceptedCmp4', 'AcceptedCmp5', 'AcceptedCmp1',
       'AcceptedCmp2', 'Complain', 'Z_CostContact', 'Z_Revenue', 'Response',
       'Age', 'Customer_Days', 'marital_Divorced', 'marital_Married',
       'marital_Single', 'marital_Together', 'marital_Widow',
       'education_2n Cycle', 'education_Basic', 'education_Graduation',
       'education_Master', 'education_PhD', 'MntTotal', 'MntRegularProds',
       'AcceptedCmpOverall'],
      dtype='object')
```

In [5]:

```
# Display the first 5 rows of the pandas DataFrame to preview the data
data.head()
```

Out[5]:

	Income	Kidhome	Teenhome	Recency	MntWines	MntFruits	MntMeatProducts	MntFishProdu
0	58138.0	0	0	58	635	88	546	
1	46344.0	1	1	38	11	1	6	
2	71613.0	0	0	26	426	49	127	
3	26646.0	1	0	26	11	4	20	
4	58293.0	1	0	94	173	43	118	

5 rows × 39 columns

In [6]:

```
# Display the last 5 rows of the pandas DataFrame to preview the data
data.tail()
```

Out[6]:

	Income	Kidhome	Teenhome	Recency	MntWines	MntFruits	MntMeatProducts	MntFishPr
2200	61223.0	0	1	46	709	43	182	
2201	64014.0	2	1	56	406	0	30	
2202	56981.0	0	0	91	908	48	217	
2203	69245.0	0	1	8	428	30	214	
2204	52869.0	1	1	40	84	3	61	

5 rows × 39 columns

In [7]:

```
# some information about data
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2205 entries, 0 to 2204
Data columns (total 39 columns):
 #   Column                Non-Null Count  Dtype  
---  ------                --------------  -----  
 0   Income                2205 non-null   float64
 1   Kidhome               2205 non-null   int64  
 2   Teenhome              2205 non-null   int64  
 3   Recency               2205 non-null   int64  
 4   MntWines              2205 non-null   int64  
 5   MntFruits             2205 non-null   int64  
 6   MntMeatProducts       2205 non-null   int64  
 7   MntFishProducts       2205 non-null   int64  
 8   MntSweetProducts      2205 non-null   int64  
 9   MntGoldProds          2205 non-null   int64  
 10  NumDealsPurchases     2205 non-null   int64  
 11  NumWebPurchases       2205 non-null   int64  
 12  NumCatalogPurchases   2205 non-null   int64  
 13  NumStorePurchases     2205 non-null   int64  
 14  NumWebVisitsMonth     2205 non-null   int64  
 15  AcceptedCmp3          2205 non-null   int64  
 16  AcceptedCmp4          2205 non-null   int64  
 17  AcceptedCmp5          2205 non-null   int64  
 18  AcceptedCmp1          2205 non-null   int64  
 19  AcceptedCmp2          2205 non-null   int64  
 20  Complain              2205 non-null   int64  
 21  Z_CostContact         2205 non-null   int64  
 22  Z_Revenue             2205 non-null   int64  
 23  Response              2205 non-null   int64  
 24  Age                   2205 non-null   int64  
 25  Customer_Days         2205 non-null   int64  
 26  marital_Divorced      2205 non-null   int64  
 27  marital_Married       2205 non-null   int64  
 28  marital_Single        2205 non-null   int64  
 29  marital_Together      2205 non-null   int64  
 30  marital_Widow         2205 non-null   int64  
 31  education_2n Cycle    2205 non-null   int64  
 32  education_Basic       2205 non-null   int64  
 33  education_Graduation  2205 non-null   int64  
 34  education_Master      2205 non-null   int64  
 35  education_PhD         2205 non-null   int64  
 36  MntTotal              2205 non-null   int64  
 37  MntRegularProds       2205 non-null   int64  
 38  AcceptedCmpOverall    2205 non-null   int64  
dtypes: float64(1), int64(38)
memory usage: 672.0 KB
```

In [8]:

```
# some statistical information about data
data.describe()
```

Out[8]:

	Income	Kidhome	Teenhome	Recency	MntWines	MntFruits	MntMeat
count	2205.000000	2205.000000	2205.000000	2205.000000	2205.000000	2205.000000	220
mean	51622.094785	0.442177	0.506576	49.009070	306.164626	26.403175	16
std	20713.063826	0.537132	0.544380	28.932111	337.493839	39.784484	21
min	1730.000000	0.000000	0.000000	0.000000	0.000000	0.000000	
25%	35196.000000	0.000000	0.000000	24.000000	24.000000	2.000000	1
50%	51287.000000	0.000000	0.000000	49.000000	178.000000	8.000000	6
75%	68281.000000	1.000000	1.000000	74.000000	507.000000	33.000000	23
max	113734.000000	2.000000	2.000000	99.000000	1493.000000	199.000000	172

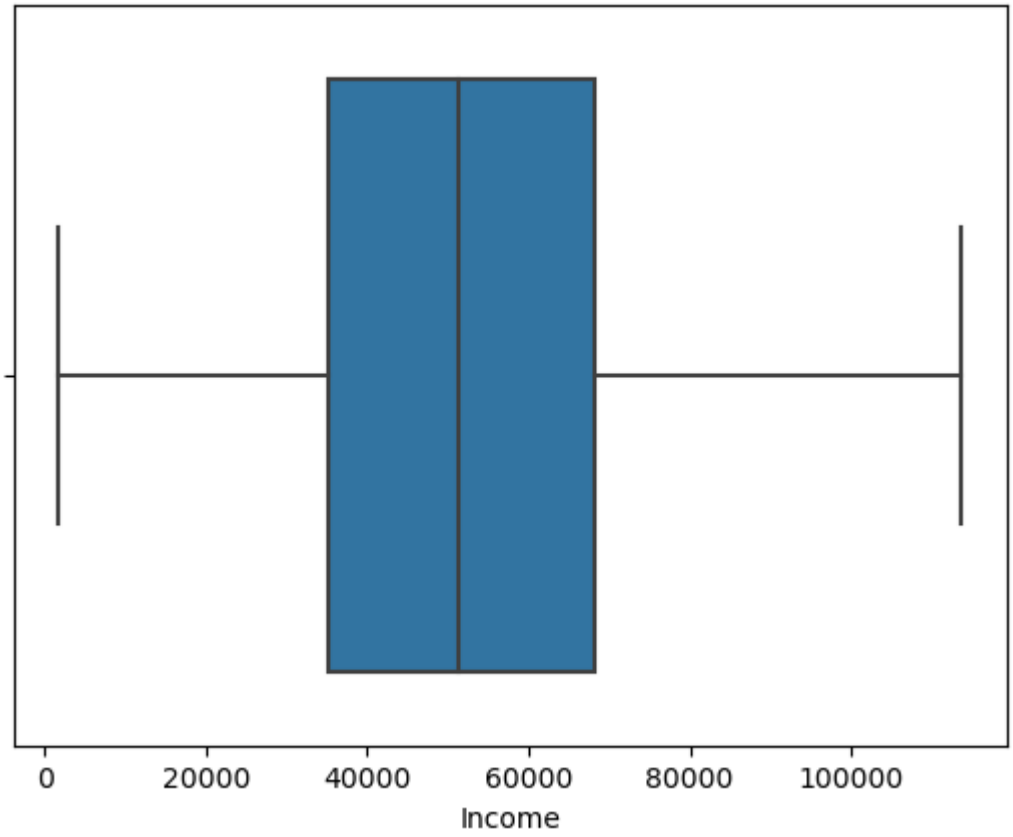
8 rows × 39 columns

In [9]:

```
# Create a box plot of the "Income" column using seaborn
sns.boxplot(x=data["Income"])
```

Out[9]:

```
<Axes: xlabel='Income'>
```



In [10]:

```
# Calculate the third and first quartiles of the "Income" column using numpy
q3, q1 = np.percentile(data['Income'], [75, 25])
```

In [11]:

```
# 75% from data
q3
```

Out[11]:

```
68281.0
```

In [12]:

```
# 25% from data
q1
```

Out[12]:

```
35196.0
```

In [13]:

```
# Calculate the interquartile range (IQR) of the "Income" column
iqr = q3 - q1
iqr
```

Out[13]:

```
33085.0
```

first range from 1730 to 35196

second range from 35196 to 68281

third range from 68281 to 113734

In [14]:

```
# Create a numpy array of the values in the "Income" column of the pandas DataFrame
Income_array = data['Income'].values
```

In [15]:

```
# display the array
Income_array
```

Out[15]:

```
array([58138., 46344., 71613., ..., 56981., 69245., 52869.])
```

In [16]:

```
counter_low = 0
for i_low in Income_array:
    if (i_low >=1730) & (i_low < 35196):
        counter_low +=1
    else:
        continue
```

In [17]:

```
# number of customers whose income is low (between 1730 and 35196)
counter_low
```

Out[17]:

```
551
```

In [18]:

```
# Count the number of values in the "Income" column that fall within a certain range from
35196 and 68280
counter_average = 0
for i_average in Income_array:
    if (i_average >= 35196) & (i_average < 68281):
        counter_average +=1
    else:
        continue
```

In [19]:

```
# number of customers whose income is average (between 35196 and 68281)
counter_average
```

Out[19]:

```
1102
```

In [20]:

```
# Count the number of values in the "Income" column that fall within a certain range from
68281 and 113734
counter_high = 0
for i_high in Income_array:
    if (i_high >= 68281) & (i_high <= 113734):
        counter_high +=1
    else:
        continue
```

In [21]:

```
# number of customers whose income is high (between 68281 and 113734)
counter_high
```

Out[21]:

```
552
```

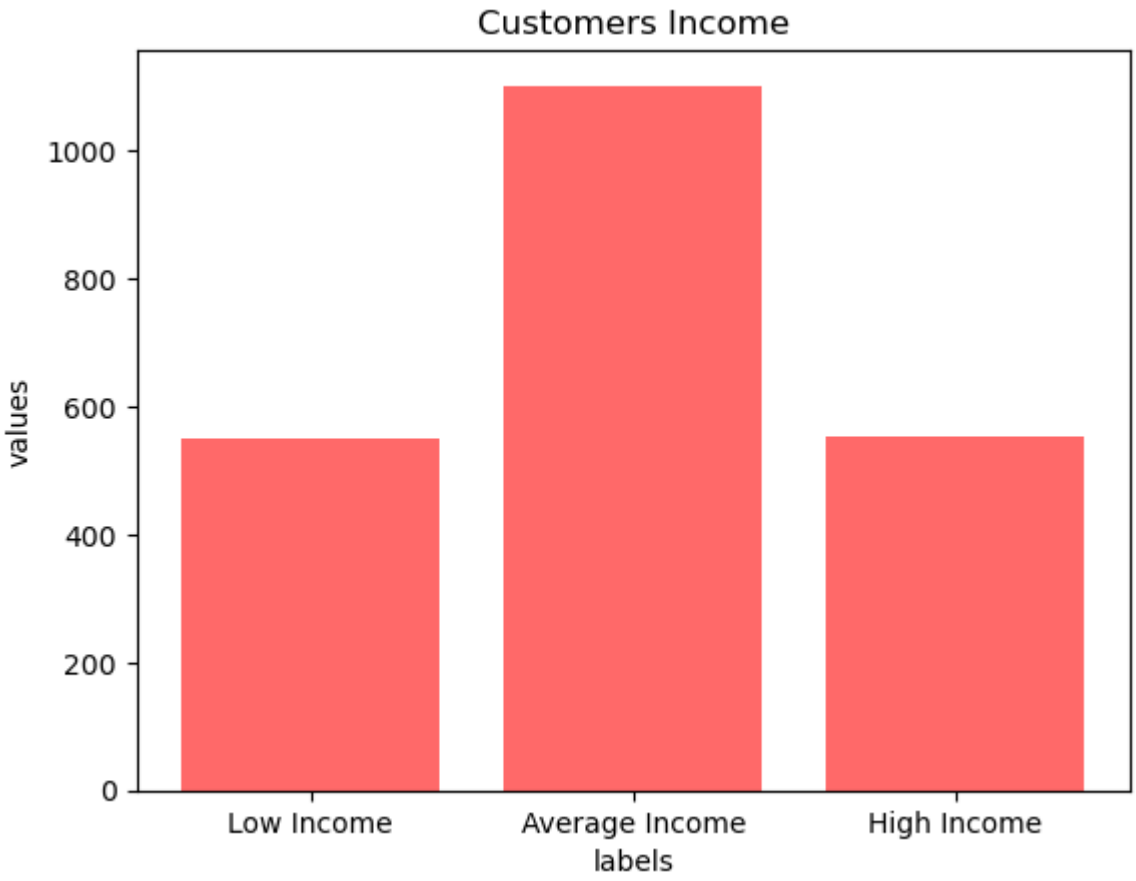
Plot for customers income

In [22]:

```
# Create a bar chart of customers' income levels
customers_Income = [551, 1102, 552]
# Define the values to be plotted
labels = ["Low Income", "Average Income", "High Income"]
 # Define the x-axis labels
plt.xticks(range(len(customers_Income)), labels)
# Set the x-tick labels
plt.xlabel('labels')
# Set the x-axis label
plt.ylabel('values')
# Set the y-axis label
plt.title('Customers Income')
# Set the title of the plot
plt.bar(range(len(customers_Income)), customers_Income, color="#FF6969")
```

Out[22]:

```
<BarContainer object of 3 artists>
```



In [23]:

```
# Create a new column in the pandas DataFrame based on a condition
# The condition checks whether the value in the "Income" column is between 1730 and 35196
(indicating a low income customer), and if so, assigns the value from the "Kidhome" column
to the new column. If the value in the "Income" column does not meet the condition, the ne
w column is assigned a value of 3.
data['kids for low income customer'] = np.where((data['Income'] >= 1730) & (data['Income']
< 35196) , data['Kidhome'] , 3)
```

In [24]:

```
# Create a numpy array of the values in a column of the pandas DataFrame
array1 = data['kids for low income customer'].values
```

In [25]:

```
# dataframe for customers whose income is low and don't have kids
df_low_zero = (data['kids for low income customer'] == 0)
data[df_low_zero]
```

Out[25]:

	Income	Kidhome	Teenhome	Recency	MntWines	MntFruits	MntMeatProducts	MntFishPr
10	7500.0	0	0	59	6	16	11	
13	17323.0	0	0	38	3	14	17	
24	18589.0	0	0	89	6	4	25	
28	10979.0	0	0	34	8	4	10	
40	21994.0	0	1	4	9	0	6	
...	...	...	...	...	...	...	...	
2159	27469.0	0	0	2	9	1	2	
2171	18929.0	0	0	15	32	0	8	
2175	14918.0	0	1	52	3	3	3	
2181	5305.0	0	1	12	12	4	7	
2198	26816.0	0	0	50	5	1	6	

134 rows × 40 columns

In [26]:

```
# groups the pandas DataFrame object named "data" by the "kids for low income customer" co
lumn, and calculates the sum of the "AcceptedCmp1" through "AcceptedCmp5" columns for each
group. The resulting output is a new pandas DataFrame named "totalkidsAcceptedCmp".
totalkidsAcceptedCmp = data.groupby("kids for low income customer")[['AcceptedCmp1', 'Acce
ptedCmp2', 'AcceptedCmp3', 'AcceptedCmp4', 'AcceptedCmp5']].sum()
```

Number of customers who accept the offer in each campaign and their income is low

In [27]:

```
# Display the pandas DataFrame showing the total accepted campaigns for each group of low income customers
totalkidsAcceptedCmp
```

Out[27]:

kids for low income customer	AcceptedCmp1	AcceptedCmp2	AcceptedCmp3	AcceptedCmp4	AcceptedCmp5
0	0	0	6	0	0
1	0	0	37	1	0
2	0	0	0	0	0
3	142	30	120	163	161

How many customers with low income who don\'t have kids and accepted the offer in each campaign ?

The number of low-income customers who don't have kids and accepted the offer in the first campaign is zero

The number of low-income customers who don't have kids and accepted the offer in the second campaign is zero

The number of low-income customers who don't have kids and accepted the offer in the third campaign is 6

The number of low-income customers who don't have kids and accepted the offer in the fourth campaign is zero

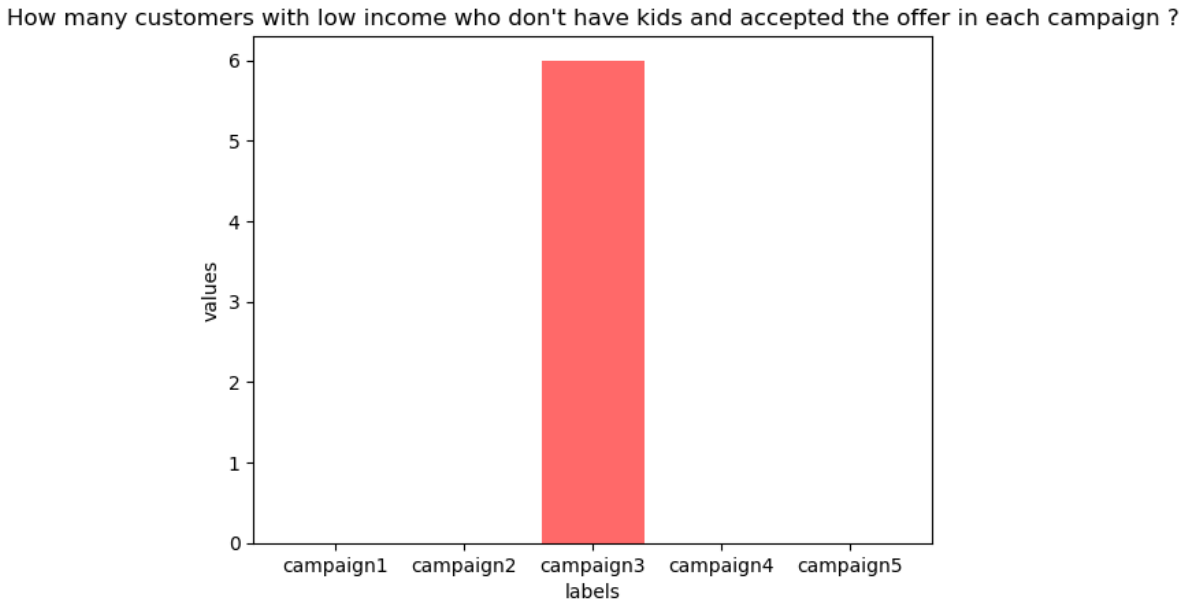
The number of low-income customers who don't have kids and accepted the offer in the Fifth campaign is zero

In [28]:

```
# Create a bar chart showing the number of low income customers who accepted each campaign offer
low_income_customers_campaigns = [0, 0, 6, 0, 0]
# Define the values to be plotted
labels = ["campaign1", "campaign2", "campaign3", "campaign4", "campaign5"]
# Define the x-axis labels
plt.xticks(range(len(low_income_customers_campaigns)), labels)
# Set the x-tick labels
plt.xlabel('labels')
# Set the y-axis label
plt.ylabel('values')
# Set the title of the plot
plt.title('How many customers with low income who don\'t have kids and accepted the offer in each campaign ?')
# Plot the bar chart
plt.bar(range(len(low_income_customers_campaigns)), low_income_customers_campaigns, color ="#FF6969")
```

Out[28]:

```
<BarContainer object of 5 artists>
```



In [29]:

```
# checks whether the value in the "kids for low income customer" column of the pandas Data Frame named "data" is equal to 1. The code then filters the DataFrame to show only the rows where the condition is true
df_low_one = (data['kids for low income customer'] == 1)
data[df_low_one]
```

Out[29]:

	Income	Kidhome	Teenhome	Recency	MntWines	MntFruits	MntMeatProducts	MntFishPr
3	26646.0	1	0	26	11	4	20	
7	33454.0	1	0	32	76	10	56	
8	30351.0	1	0	19	14	0	24	
9	5648.0	1	1	68	28	0	6	
18	33812.0	1	0	86	4	17	19	
...	...	...	...	...	...	...	...	
2185	22775.0	1	0	40	5	1	8	
2189	7500.0	1	0	7	2	8	11	
2190	33562.0	1	2	33	21	12	12	
2196	11012.0	1	0	82	24	3	26	
2199	34421.0	1	0	81	3	3	7	

398 rows × 40 columns

How many customers with low income who have one kid and accepted the offer in each campaign ?

The number of low-income customers who have one kid and accepted the offer in the first campaign is zero

The number of low-income customers who have one kid and accepted the offer in the second campaign is zero

The number of low-income customers who have one kid and accepted the offer in the third campaign is 37

The number of low-income customers who have one kid and accepted the offer in the fourth campaign is 1

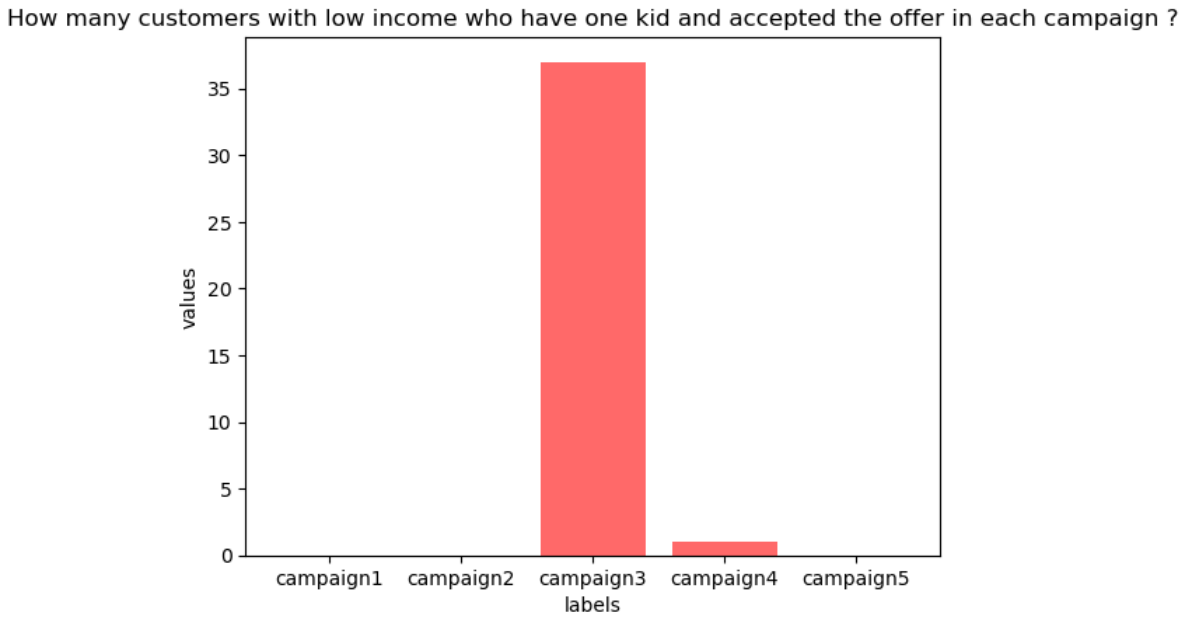
The number of low-income customers who have one kid and accepted the offer in the fifth campaign is zero

In [30]:

```
# Create a bar chart showing the number of low income customers with one kid who accepted
each campaign offer
low_income_customers_campaigns_one_kid = [0, 0, 37, 1, 0]
# Define the values to be plotted
labels = ["campaign1", "campaign2", "campaign3", "campaign4", "campaign5"]
# Define the x-axis labels
plt.xticks(range(len(low_income_customers_campaigns_one_kid)), labels)
# Set the x-axis label
plt.xlabel('labels')
# Set the y-axis label
plt.ylabel('values')
# Set the title of the plot
plt.title('How many customers with low income who have one kid and accepted the offer in e
ach campaign ?')
# Plot the bar chart
plt.bar(range(len(low_income_customers_campaigns_one_kid)), low_income_customers_campaigns
_one_kid, color="#FF6969")
```

Out[30]:

```
<BarContainer object of 5 artists>
```



In [31]:

```
# Checks whether the value in the "kids for low income customer" column of the pandas Data Frame named "data" is equal to 2. The code then filters the DataFrame to show only the rows where the condition is true
df_low_two = (data['kids for low income customer'] == 2)
data[df_low_two]
```

Out[31]:

	Income	Kidhome	Teenhome	Recency	MntWines	MntFruits	MntMeatProducts	MntFishPr
113	19510.0	2	0	63	9	0	7	
126	33762.0	2	1	61	53	1	34	
149	30523.0	2	1	0	5	0	3	
279	16626.0	2	0	76	8	3	22	
297	22574.0	2	1	28	25	0	8	
345	28442.0	2	0	53	19	3	10	
363	33581.0	2	0	38	11	0	5	
590	26150.0	2	1	61	5	1	13	
596	22574.0	2	1	28	25	0	8	
656	26751.0	2	0	26	1	1	5	
829	15072.0	2	0	96	8	2	15	
847	19485.0	2	0	80	6	0	4	
1395	34578.0	2	1	1	7	0	1	
1533	27215.0	2	1	50	30	5	22	
1555	34633.0	2	1	31	8	1	5	
1635	34916.0	2	0	89	51	23	82	
1907	29543.0	2	0	47	17	3	18	
2097	33590.0	2	1	65	4	0	2	
2195	24434.0	2	0	9	3	2	8	

19 rows × 40 columns

How many customers with low income who have two kids and accepted the offer in each campaign ?

The number of low-income customers who have two kids and accepted the offer in the first campaign is zero

The number of low-income customers who have two kids and accepted the offer in the second campaign is zero

The number of low-income customers who have two kids and accepted the offer in the third campaign is zero

The number of low-income customers who have two kids and accepted the offer in the fourth campaign is zero

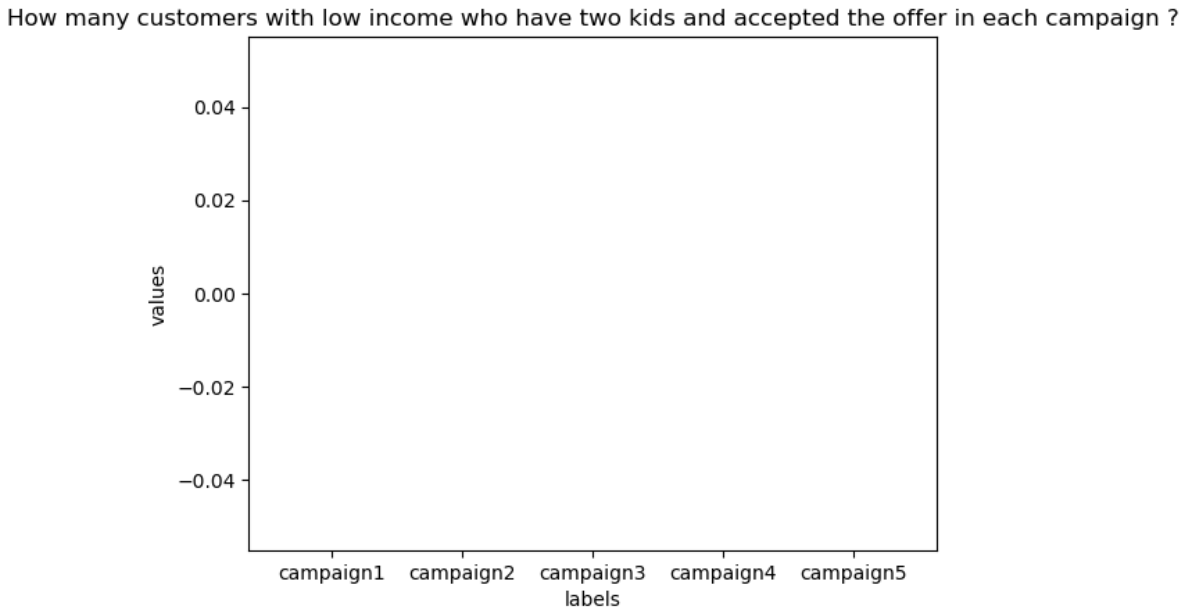
The number of low-income customers who have two kids and accepted the offer in the fifth campaign is zero

In [32]:

```
# Create a bar chart showing the number of low income customers with two kids who accepted
each campaign offe
low_income_customers_campaigns_two_kids = [0, 0, 0, 0, 0]
# Define the values to be plotted
labels = ["campaign1", "campaign2", "campaign3", "campaign4", "campaign5"]
# Set the x-tick labels
plt.xticks(range(len(low_income_customers_campaigns_two_kids)), labels)
# Set the x-axis label
plt.xlabel('labels')
# Set the y-axis label
plt.ylabel('values')
# Set the title of the plot
plt.title('How many customers with low income who have two kids and accepted the offer in
each campaign ?')
# Plot the bar chart
plt.bar(range(len(low_income_customers_campaigns_two_kids)), low_income_customers_campaign
s_two_kids, color="#FF6969")
```

Out[32]:

```
<BarContainer object of 5 artists>
```



In [33]:

```
# Count the number of low income customers with zero kids
low_ZeroKids = 0
for count_low0 in array1:
    if count_low0 == 0:
        low_ZeroKids+=1
    else:
        continue
```

In [34]:

```
# display number of customer's low income with 0 kids
print(low_ZeroKids)
```

```
134
```

In [35]:

```
# Count the number of low income customers with one kid
low_oneKid = 0
for count_low1 in array1:
    if count_low1 == 1:
        low_oneKid+=1
    else:
        continue
```

In [36]:

```
# display number of customer's low income with 1 kids
print(low_oneKid)
```

```
398
```

In [37]:

```
# Count the number of low income customers with two kids
low_twoKids = 0
for count_low2 in array1:
    if count_low2 == 2:
        low_twoKids+=1
    else:
        continue
```

In [38]:

```
# display number of customer's low income with 2 kids
print(low_twoKids)
```

```
19
```

In [39]:

```
data.shape
```

Out[39]:

```
(2205, 40)
```

Number of kids for each customer based on their low income

24% from Customers whose income is low do not have kids in their home

72% from Customers whose income is low have one kid in their home

3% from Customers whose income is low have two kids in their home

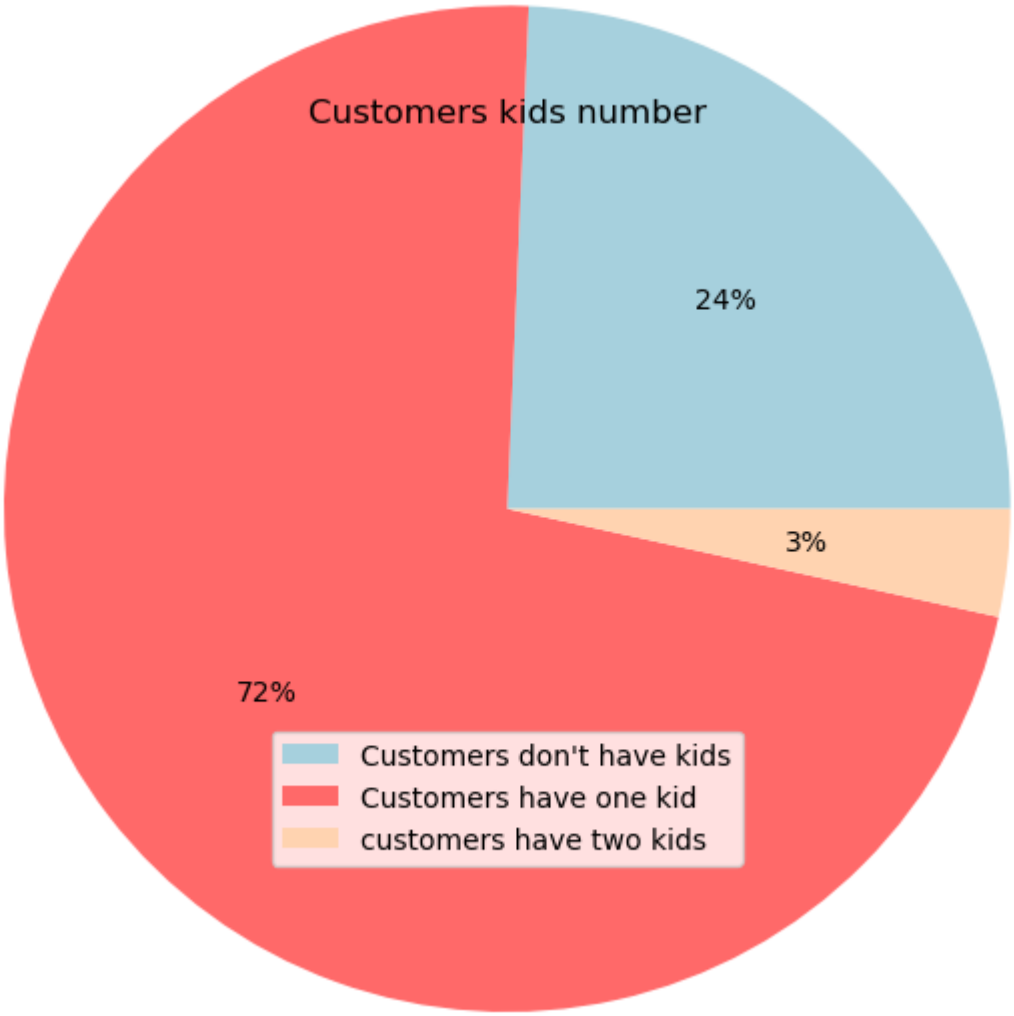
In [40]:

```
# Create a pie chart showing the number of low income customers with different numbers of
kids
low_income_kids =[134, 398, 19] # Define the values to be plotted

# Plot the pie chart with appropriate attributes
plt.pie(low_income_kids,  radius=1.7, colors=["#A6D0DD","#FF6969", "#FFD3B0"],
        rotatelabels=False, autopct= "%1.0f%%")
# Set the title of the plot
plt.title('Customers kids number')
# Add the legend to the plot
plt.legend(labels=["Customers don't have kids", "Customers have one kid", "customers have
two kids"])
```

Out[40]:

```
<matplotlib.legend.Legend at 0x780006c390f0>
```



In [41]:

```
# checks whether the value in the "Income" column is greater than or equal to 35196 and le
ss than 68281. If the condition is true, the value in the "Kidhome" column is used for the
new column. Otherwise, the value 3 is used for the new column.
data['kids for average income customer'] = np.where((data['Income'] >= 35196) & (data['Inc
ome'] < 68281) , data['Kidhome'] , 3)
```

In [42]:

```
# Convert a pandas DataFrame column to a numpy array
array2 = data['kids for average income customer'].values
```

In [43]:

```
# checks whether the value in the "kids for average income customer" column of the pandas
DataFrame named "data" is equal to 0.
df_average_zero = (data['kids for average income customer'] == 0)
data[df_average_zero]
```

Out[43]:

	Income	Kidhome	Teenhome	Recency	MntWines	MntFruits	MntMeatProducts	MntFishPr
0	58138.0	0	0	58	635	88	546	
5	62513.0	0	1	16	520	42	98	
6	55635.0	0	1	34	235	65	164	
11	63033.0	0	0	82	194	61	480	
16	37760.0	0	0	20	84	5	38	
...	...	...	...	...	...	...	...	
2191	57642.0	0	1	24	580	6	58	
2194	57967.0	0	1	39	229	7	137	
2197	44802.0	0	0	71	853	10	143	
2200	61223.0	0	1	46	709	43	182	
2202	56981.0	0	0	91	908	48	217	

630 rows × 41 columns

In [44]:

```
# groups the pandas DataFrame object named "data" by the "kids for average income custome
r" column, and calculates the sum of the "AcceptedCmp1" through "AcceptedCmp5" columns for
each group. The resulting output is a new pandas DataFrame named "averageIncome_totalKidsA
cceptedCmp"
averageIncome_totalKidsAcceptedCmp = data.groupby("kids for average income customer")[['Ac
ceptedCmp1', 'AcceptedCmp2', 'AcceptedCmp3', 'AcceptedCmp4', 'AcceptedCmp5']].sum()
```

Number of customers who accept the offer in each campaign and their income is average based on number of kids

In [45]:

```
# Display a pandas DataFrame object showing the total number of accepted campaigns for eac
h category of a categorical variable
averageIncome_totalKidsAcceptedCmp
```

Out[45]:

kids for average income customer	AcceptedCmp1	AcceptedCmp2	AcceptedCmp3	AcceptedCmp4	AcceptedCmp5
0	17	12	41	72	9
1	2	2	35	14	0
2	2	0	1	0	0
3	121	16	86	78	152

How many customers with average income who don't have kids and accepted the offer in each campaign ?

The number of average-income customers who don't have kids and accepted the offer in the first campaign is 17

The number of average-income customers who don't have kids and accepted the offer in the second campaign is 12

The number of average-income customers who don't have kids and accepted the offer in the third campaign is 41

The number of average-income customers who don't have kids and accepted the offer in the fourth campaign is 72

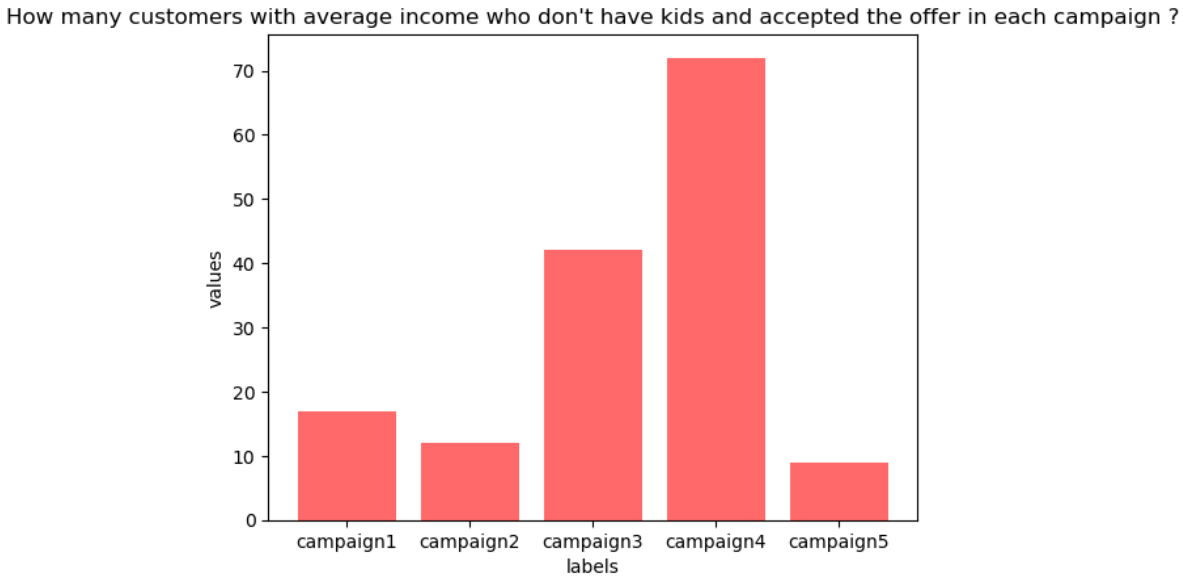
The number of average-income customers who don't have kids and accepted the offer in the fifth campaign is 9

In [46]:

```
# Create a bar chart showing the number of average income customers with no kids who accep
ted each campaign offe
average_income_customers_campaigns_zero_kid = [17, 12, 42, 72, 9]
# Define the x-axis labels
labels = ["campaign1", "campaign2", "campaign3", "campaign4", "campaign5"]
# Set the x-tick labels
plt.xticks(range(len(average_income_customers_campaigns_zero_kid)), labels)
# Set the x-axis label
plt.xlabel('labels')
# Set the y-axis label
plt.ylabel('values')
# Set the title of the plot
plt.title('How many customers with average income who don\'t have kids and accepted the of
fer in each campaign ?')
# Plot the bar chart
plt.bar(range(len(average_income_customers_campaigns_zero_kid)), average_income_customers_
campaigns_zero_kid, color="#FF6969")
```

Out[46]:

```
<BarContainer object of 5 artists>
```



In [47]:

```
# checks whether the value in the "kids for average income customer" column of the pandas DataFrame named "data" is equal to 1.
df_average_one = (data['kids for average income customer'] == 1)
data[df_average_one]
```

Out[47]:

	Income	Kidhome	Teenhome	Recency	MntWines	MntFruits	MntMeatProducts	MntFishPr
1	46344.0	1	1	38	11	1	6	
4	58293.0	1	0	94	173	43	118	
12	59354.0	1	1	53	233	2	53	
15	41850.0	1	1	51	53	5	19	
25	53359.0	1	1	4	173	4	30	
...	...	...	...	...	...	...	...	
2182	36807.0	1	1	88	4	2	5	
2186	40101.0	1	0	73	171	3	129	
2192	58554.0	1	1	55	368	24	68	
2193	63777.0	1	1	87	457	5	106	
2204	52869.0	1	1	40	84	3	61	

446 rows × 41 columns

How many customers with average income who have one kid and accepted the offer in each campaign ?

The number of average-income customers who have one kid and accepted the offer in the first campaign is 2

The number of average-income customers who have one kid and accepted the offer in the second campaign is 2

The number of average-income customers who have one kid and accepted the offer in the third campaign is 35

The number of average-income customers who have one kid and accepted the offer in the fourth campaign is 14

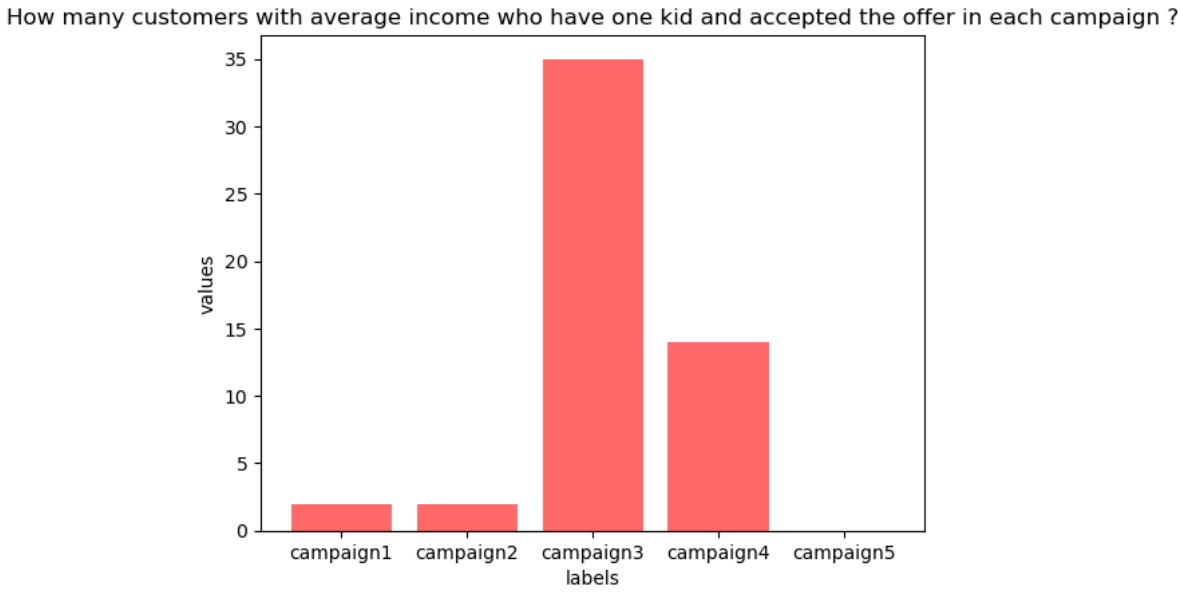
The number of average-income customers who have one kid and accepted the offer in the fifth campaign is zero

In [48]:

```
# Create a bar chart showing the number of average income customers with one kid who accep
ted each campaign offer
average_income_customers_campaigns_one_kid = [2, 2, 35, 14, 0]
# Define the values to be plotted
labels = ["campaign1", "campaign2", "campaign3", "campaign4", "campaign5"]
# Set the x-tick labels
plt.xticks(range(len(average_income_customers_campaigns_one_kid)), labels)
# Set the x-axis label
plt.xlabel('labels')
# Set the y-axis label
plt.ylabel('values')
# Set the title of the plot
plt.title('How many customers with average income who have one kid and accepted the offer
in each campaign ?')
# Plot the bar chart
plt.bar(range(len(average_income_customers_campaigns_one_kid)), average_income_customers_c
ampaigns_one_kid, color="#FF6969")
```

Out[48]:

```
<BarContainer object of 5 artists>
```



In [49]:

```
# checks whether the value in the "kids for average income customer" column of the pandas DataFrame named "data" is equal to 2
df_average_two = (data['kids for average income customer'] == 2)
data[df_average_two]
```

Out[49]:

	Income	Kidhome	Teenhome	Recency	MntWines	MntFruits	MntMeatProducts	MntFishPr
137	35688.0	2	1	94	73	3	90	
153	43482.0	2	1	83	18	1	32	
160	50447.0	2	0	4	85	7	24	
166	38285.0	2	1	96	2	0	5	
203	52195.0	2	1	2	12	0	4	
245	40737.0	2	1	24	11	0	4	
312	54432.0	2	1	37	33	0	5	
366	35688.0	2	1	94	73	3	90	
599	35791.0	2	1	94	27	0	5	
617	46102.0	2	1	3	14	0	1	
708	56962.0	2	1	60	292	3	77	
837	55357.0	2	0	66	374	64	116	
862	36627.0	2	0	78	9	1	5	
866	46231.0	2	1	87	189	2	55	
933	42767.0	2	0	53	20	6	43	
951	40706.0	2	1	37	59	0	11	
957	56962.0	2	1	60	292	3	77	
1107	46931.0	2	1	94	41	0	17	
1221	59062.0	2	1	74	46	1	12	
1302	37774.0	2	0	28	173	8	107	
1351	35791.0	2	1	94	27	0	5	
1393	36026.0	2	1	34	20	4	10	
1462	64014.0	2	1	56	406	0	30	
1551	45203.0	2	0	4	35	3	67	
1830	58217.0	2	1	84	68	1	13	
2201	64014.0	2	1	56	406	0	30	

26 rows × 41 columns

How many customers with average income who have two kids and accepted the offer in each campaign ?

The number of average-income customers who have two kids and accepted the offer in the first campaign is 2

The number of average-income customers who have two kids and accepted the offer in the second campaign is zero

The number of average-income customers who have two kids and accepted the offer in the third campaign is 1

The number of average-income customers who have two kids and accepted the offer in the fourth campaign is zero

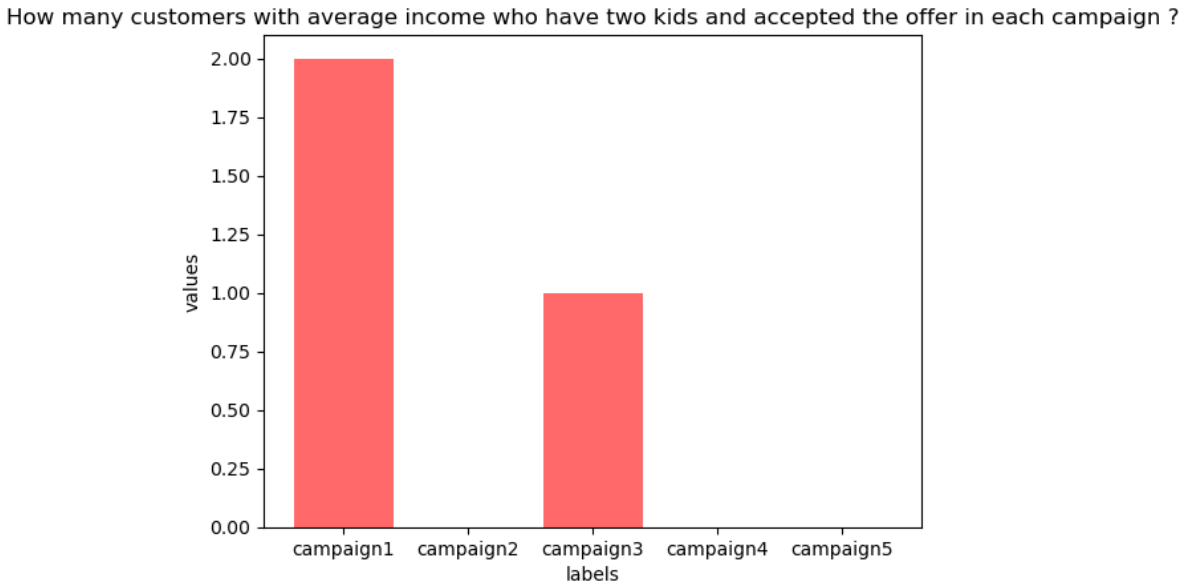
The number of average-income customers who have two kids and accepted the offer in the fifth campaign is zero

In [50]:

```
# Create a bar chart showing the number of average income customers with two kids who acce
pted each campaign offer
average_income_customers_campaigns_two_kids = [2, 0, 1, 0, 0]
# Define the x-axis labels
labels = ["campaign1", "campaign2", "campaign3", "campaign4", "campaign5"]
# Set the x-tick labels
plt.xticks(range(len(average_income_customers_campaigns_two_kids)), labels)
# Set the x-axis label
plt.xlabel('labels')
# Set the y-axis label
plt.ylabel('values')
# Set the title of the plot
plt.title('How many customers with average income who have two kids and accepted the offer
in each campaign ?')
# Plot the bar chart
plt.bar(range(len(average_income_customers_campaigns_two_kids)), average_income_customers_
campaigns_two_kids, color="#FF6969")
```

Out[50]:

```
<BarContainer object of 5 artists>
```



In [51]:

```
# Count the number of average income customers with zero kids
average_ZeroKids = 0
for count_average0 in array2:
    if count_average0 == 0:
        average_ZeroKids+=1
    else:
        continue
```

In [52]:

```
# display number of customer's average income with 0 kids
print(average_ZeroKids)
```

630

In [53]:

```
# Count the number of average income customers with one kid
average_oneKid = 0
for count_average1 in array2:
    if count_average1 == 1:
        average_oneKid+=1
    else:
        continue
```

In [54]:

```
# display number of customer's average income with 1 kid
print(average_oneKid)
```

446

In [55]:

```
# Count the number of average income customers with two kids
average_twoKids = 0
for count_average2 in array2:
    if count_average2 == 2:
        average_twoKids+=1
    else:
        continue
```

In [56]:

```
# display number of customer's average income with 2 kids
print(average_twoKids)
```

26

Number of kids for each customer based on their average income

57% from Customers whose income is average do not have kids in their home

40% from Customers whose income is average have one kid in their home

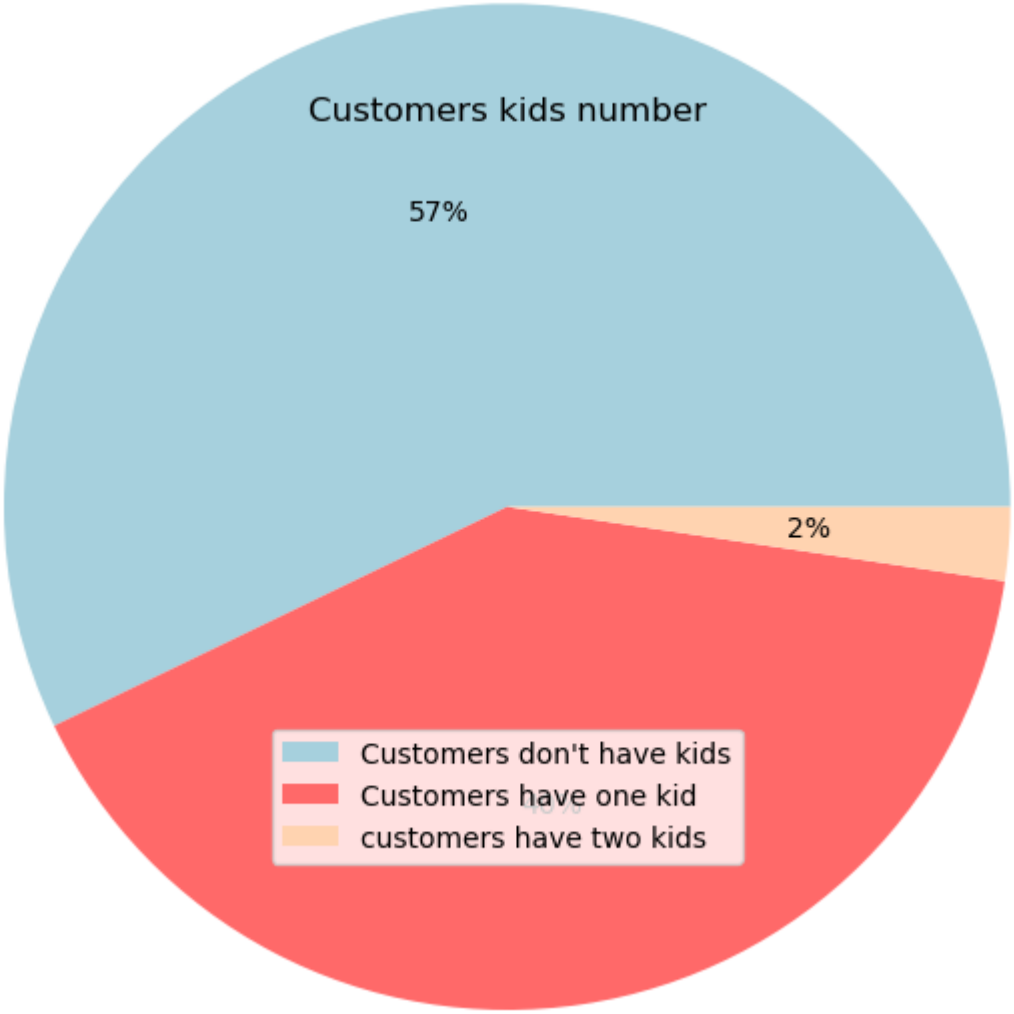
2% from Customers whose income is average have two kids in their home

In [57]:

```
# Create a pie chart showing the number of average income customers with different numbers
of kids
# Define the values to be plotted
average_income_kids =[630, 446, 26]
# Plot the pie chart with appropriate attributes
plt.pie(average_income_kids,  radius=1.7, colors=["#A6D0DD","#FF6969", "#FFD3B0"],
        rotatelabels=False, autopct= "%1.0f%%")
# Set the title of the plot
plt.title('Customers kids number')
# Add the legend to the plot
plt.legend(labels=["Customers don't have kids", "Customers have one kid", "customers have
two kids"])
```

Out[57]:

```
<matplotlib.legend.Legend at 0x78000687ff40>
```



In [58]:

```
# checks whether the value in the "Income" column is greater than or equal to 68281 and le
ss than or equal to 113734. If the condition is true, the value in the "Kidhome" column is
used for the new column. Otherwise, the value 3 is used for the new column.
data['kids for high income customer'] = np.where((data['Income'] >= 68281) & (data['Incom
e'] <= 113734) , data['Kidhome'] , 3)
```

In [59]:

```
# Convert a pandas DataFrame column to a numpy array
array3 = data['kids for high income customer'].values
```

In [60]:

```
# checks whether the value in the "kids for high income customer" column of the pandas Dat
aFrame named "data" is equal to 0.
df_high_zero = (data['kids for high income customer'] == 0)
data[df_high_zero]
```

Out[60]:

	Income	Kidhome	Teenhome	Recency	MntWines	MntFruits	MntMeatProducts	MntFishPr
2	71613.0	0	0	26	426	49	127	
14	82800.0	0	0	23	1006	22	115	
17	76995.0	0	1	91	1012	80	498	
27	84618.0	0	0	96	684	100	801	
32	68657.0	0	0	4	482	34	471	
...	...	...	...	...	...	...	...	
2178	88325.0	0	0	42	519	71	860	
2180	80617.0	0	0	42	594	51	631	
2184	82032.0	0	0	54	332	194	377	
2188	75777.0	0	0	12	712	26	538	
2203	69245.0	0	1	8	428	30	214	

512 rows × 42 columns

In [61]:

```
# groups the pandas DataFrame named "data" by the categorical variable "kids for high inco
me customer" using the "groupby" method.
highIncome_totalKidsAcceptedCmp = data.groupby("kids for high income customer")[['Accepted
Cmp1', 'AcceptedCmp2', 'AcceptedCmp3', 'AcceptedCmp4', 'AcceptedCmp5']].sum()
```

Number of customers who accept the offer in each campaign and their income is high based on the number of kids

In [62]:

```
# Display a pandas DataFrame object showing the total number of accepted campaigns for eac
h category of a categorical variable
highIncome_totalKidsAcceptedCmp
```

Out[62]:

kids for high income customer	AcceptedCmp1	AcceptedCmp2	AcceptedCmp3	AcceptedCmp4	AcceptedCmp5
0	115	16	40	70	144
1	6	0	3	7	8
2	0	0	0	0	0
3	21	14	120	87	9

How many customers with high income who don't have kids and accepted the offer in each campaign ?

The number of high-income customers who don't have kids and accepted the offer in the first campaign is 115

The number of high-income customers who don't have kids and accepted the offer in the second campaign is 16

The number of high-income customers who don't have kids and accepted the offer in the third campaign is 40

The number of high-income customers who don't have kids and accepted the offer in the fourth campaign is 70

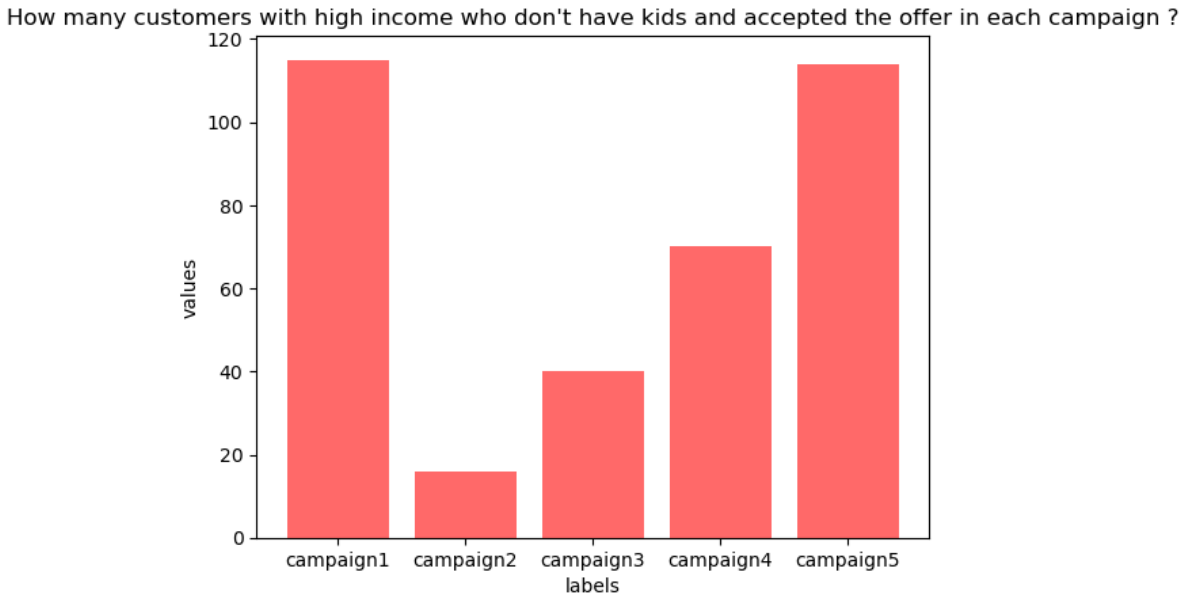
The number of high-income customers who don't have kids and accepted the offer in the fifth campaign is 144

In [63]:

```
# Create a bar chart showing the number of high income customers with zero kids who accept
ed each campaign offer
high_income_customers_campaigns_zero_kids = [115, 16, 40, 70, 114]
# Define the x-axis labels
labels = ["campaign1", "campaign2", "campaign3", "campaign4", "campaign5"]
# Set the x-tick labels
plt.xticks(range(len(high_income_customers_campaigns_zero_kids)), labels)
# Set the x-axis label
plt.xlabel('labels')
# Set the y-axis label
plt.ylabel('values')
# Set the title of the plot
plt.title('How many customers with high income who don\'t have kids and accepted the offer
in each campaign ?')
# Plot the bar chart
plt.bar(range(len(high_income_customers_campaigns_zero_kids)), high_income_customers_campa
igns_zero_kids, color="#FF6969")
```

Out[63]:

```
<BarContainer object of 5 artists>
```



In [64]:

```
# checks whether the value in the "kids for high income customer" column of the pandas Dat
aFrame named "data" is equal to 1.
df_high_one = (data['kids for high income customer'] == 1)
data[df_high_one]
```

Out[64]:

	Income	Kidhome	Teenhome	Recency	MntWines	MntFruits	MntMeatProducts	MntFishPr
45	72550.0	1	1	39	826	50	317	
64	74854.0	1	2	90	856	59	487	
119	77376.0	1	1	72	492	19	110	
206	69508.0	1	0	48	824	32	162	
232	80134.0	1	0	40	1218	16	272	
240	75702.0	1	1	77	650	28	353	
334	71952.0	1	0	93	656	80	455	
372	83664.0	1	1	57	866	21	151	
410	75433.0	1	0	28	800	0	297	
436	70829.0	1	1	87	141	70	106	
612	69084.0	1	0	43	1181	107	199	
685	71952.0	1	0	93	656	80	455	
714	89694.0	1	1	22	1126	28	211	
741	79146.0	1	1	33	245	16	223	
806	93404.0	1	2	97	1279	15	287	
904	83033.0	1	0	82	812	99	431	
1106	86358.0	1	1	78	957	47	494	
1177	70091.0	1	0	11	964	34	137	
1313	88097.0	1	0	24	163	0	480	
1386	80134.0	1	0	40	1218	16	272	
1409	70844.0	1	1	16	129	26	67	
1516	74881.0	1	1	48	505	72	270	
1544	75283.0	1	2	26	733	9	180	
1573	76445.0	1	0	2	739	107	309	
1619	75774.0	1	0	27	340	21	134	
1642	69084.0	1	0	43	1181	107	199	
1702	74881.0	1	1	48	505	72	270	
1710	79146.0	1	1	33	245	16	223	
1733	77981.0	1	0	78	138	120	204	
1773	75774.0	1	0	27	340	21	134	
1837	70886.0	1	0	65	407	70	239	
1853	70300.0	1	0	89	1045	61	338	
1862	76532.0	1	1	38	355	30	177	

	Income	Kidhome	Teenhome	Recency	MntWines	MntFruits	MntMeatProducts	MntFishPr
1898	80982.0	1	1	48	505	137	401	
2010	75330.0	1	1	94	555	82	257	
2015	83273.0	1	2	98	433	89	650	
2022	76467.0	1	0	44	676	161	426	
2030	71128.0	1	0	80	958	159	447	
2066	81929.0	1	0	60	1486	55	278	

39 rows × 42 columns

How many customers with high income who have one kid and accepted the offer in each campaign ?

The number of high-income customers who have one kid and accepted the offer in the first campaign is 6

The number of high-income customers who have one kid and accepted the offer in the second campaign is 0

The number of high-income customers who have one kid and accepted the offer in the third campaign is 3

The number of high-income customers who have one kid and accepted the offer in the fourth campaign is 7

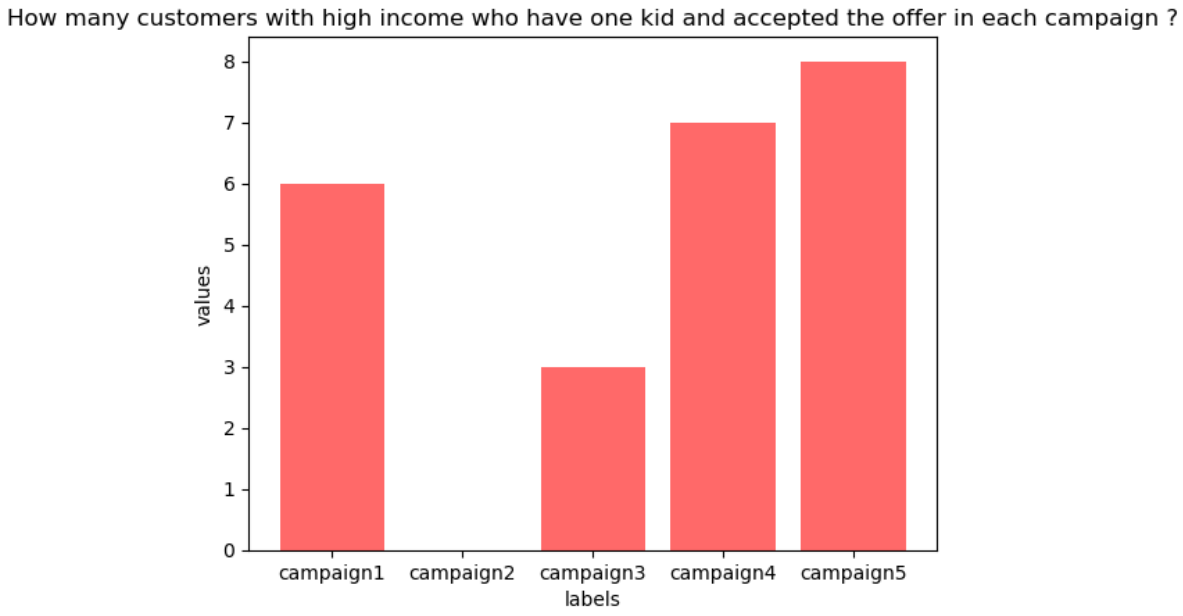
The number of high-income customers who have one kid and accepted the offer in the fifth campaign is 8

In [65]:

```
# Create a bar chart showing the number of high income customers with one kid who accepted
each campaign offer
high_income_customers_campaigns_one_kid = [6, 0, 3, 7, 8]
# Define the x-axis labels
labels = ["campaign1", "campaign2", "campaign3", "campaign4", "campaign5"]
# Set the x-tick labels
plt.xticks(range(len(high_income_customers_campaigns_one_kid)), labels)
# Set the x-axis label
plt.xlabel('labels')
# Set the y-axis label
plt.ylabel('values')
# Set the title of the plot
plt.title('How many customers with high income who have one kid and accepted the offer in
each campaign ?')
# Plot the bar chart
plt.bar(range(len(high_income_customers_campaigns_one_kid)), high_income_customers_campaig
ns_one_kid, color="#FF6969")
```

Out[65]:

```
<BarContainer object of 5 artists>
```



In [66]:

```
# checks whether the value in the "kids for high income customer" column of the pandas Dat
aFrame named "data" is equal to 2.
df_high_two = (data['kids for high income customer'] == 2)
data[df_high_two]
```

Out[66]:

	Income	Kidhome	Teenhome	Recency	MntWines	MntFruits	MntMeatProducts	MntFishPro
681	71427.0	2	0	26	212	123	177	

1 rows × 42 columns

How many customers with high income who have two kids and accepted the offer in each campaign ?

The number of high-income customers who have two kids and accepted the offer in the first campaign is zero

The number of high-income customers who have two kids and accepted the offer in the second campaign is zero

The number of high-income customers who have two kids and accepted the offer in the third campaign is zero

The number of high-income customers who have two kids and accepted the offer in the fourth campaign is zero

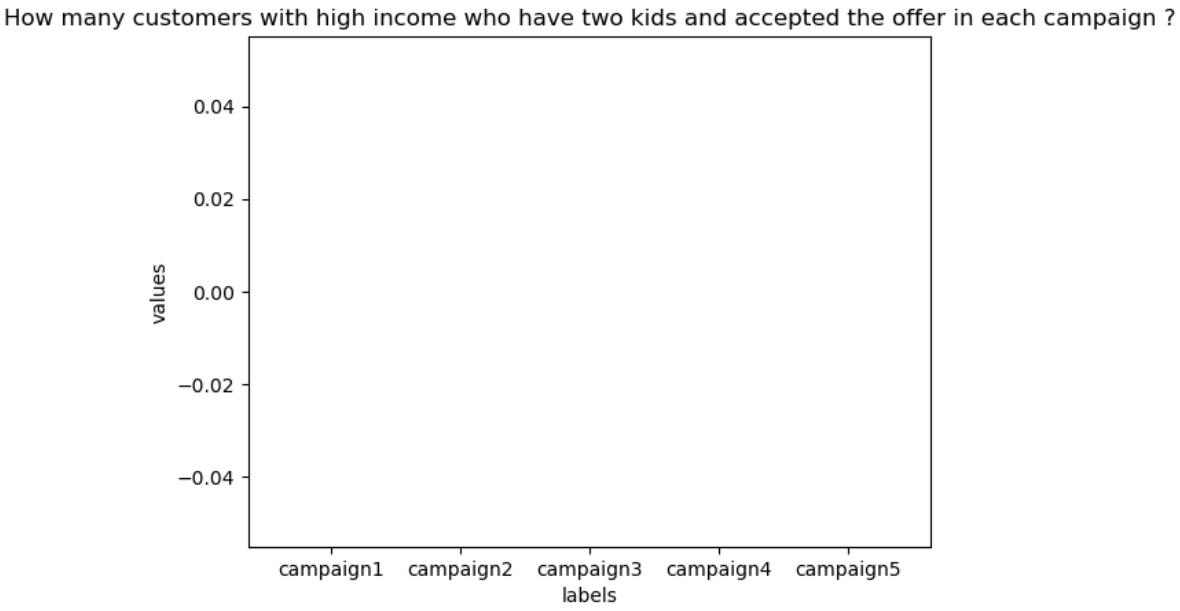
The number of high-income customers who have two kids and accepted the offer in the fifth campaign is zero

In [67]:

```
# Create a bar chart showing the number of high income customers with two kids who accepte
d each campaign offer
high_income_customers_campaigns_two_kids = [0, 0, 0, 0, 0]
# Define the x-axis labels
labels = ["campaign1", "campaign2", "campaign3", "campaign4", "campaign5"]
# Set the x-tick labels
plt.xticks(range(len(high_income_customers_campaigns_two_kids)), labels)
# Set the x-axis label
plt.xlabel('labels')
# Set the y-axis label
plt.ylabel('values')
# Set the title of the plot
plt.title('How many customers with high income who have two kids and accepted the offer in
each campaign ?')
# Plot the bar chart
plt.bar(range(len(high_income_customers_campaigns_two_kids)), high_income_customers_campai
gns_two_kids, color="#FF6969")
```

Out[67]:

```
<BarContainer object of 5 artists>
```



In [68]:

```
# Count the number of high income customers with zero kids
high_ZeroKids = 0
for count_high0 in array3:
    if count_high0 == 0:
        high_ZeroKids+=1
    else:
        continue
```

In [69]:

```
# display number of customer's high income with 0 kids
print(high_ZeroKids)
```

```
512
```

In [70]:

```
# Count the number of high income customers with one kid
high_oneKid = 0
for count_high1 in array3:
    if count_high1 == 1:
        high_oneKid+=1
    else:
        continue
```

In [71]:

```
# display number of customer's high income with 1 kid
print(high_oneKid)
```

```
39
```

In [72]:

```
# Count the number of high income customers with two kids
high_twoKids = 0
for count_high2 in array3:
    if count_high2 == 2:
        high_twoKids+=1
    else:
        continue
```

In [73]:

```
# display number of customer's high income with 2 kids
print(high_twoKids)
```

```
1
```

Number of kids for each customer based on their high income

93% from Customers whose income is high do not have kids in their home

7% from Customers whose income is high have one kid in their home

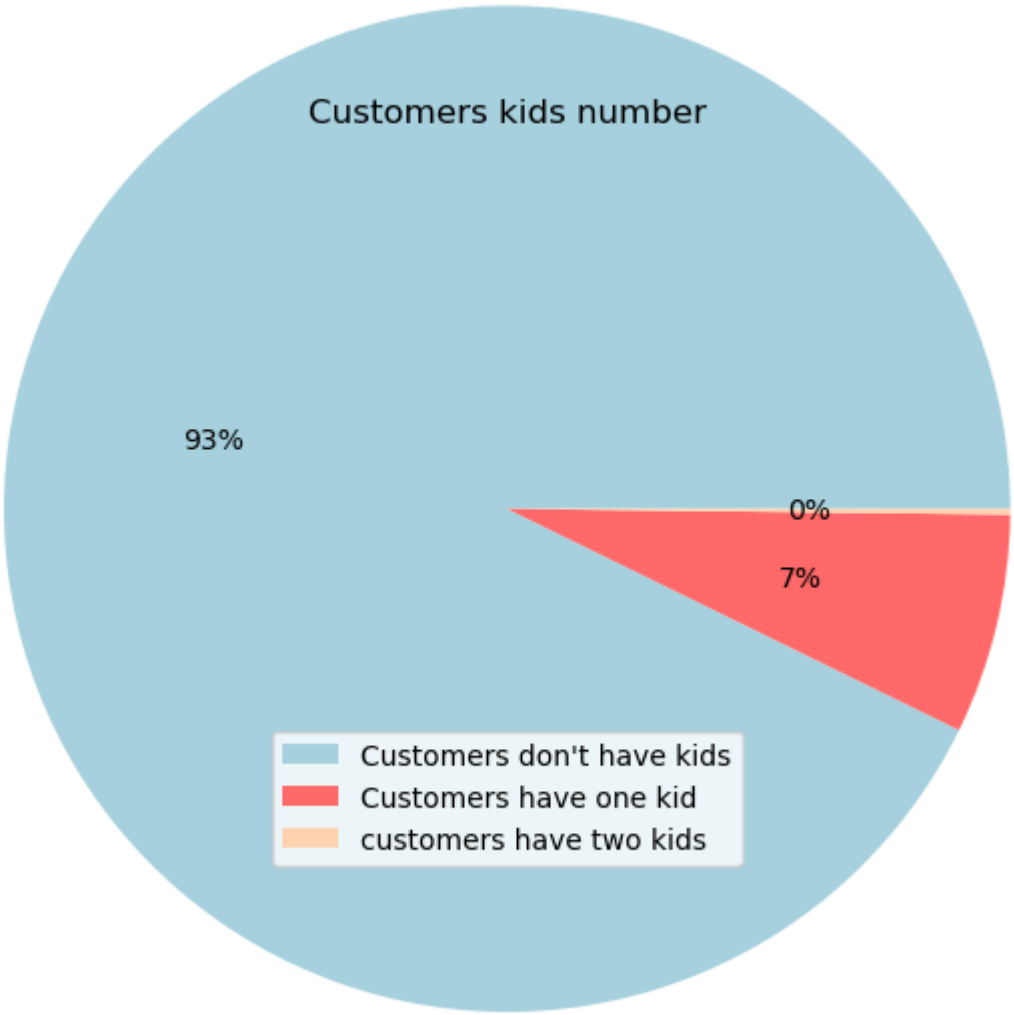
0% from Customers whose income is high have two kids in their home

In [74]:

```
# Create a pie chart showing the number of high income customers with different numbers of
kids
high_income_kids =[512, 39, 1]
# Plot the pie chart with appropriate attributes
plt.pie(high_income_kids,  radius=1.7, colors=["#A6D0DD","#FF6969", "#FFD3B0"],
        rotatelabels=False, autopct= "%1.0f%%")
# Set the title of the plot
plt.title('Customers kids number')
# Add the legend to the plot
plt.legend(labels=["Customers don't have kids", "Customers have one kid", "customers have
two kids"])
```

Out[74]:

```
<matplotlib.legend.Legend at 0x7800068577f0>
```



In [75]:

```
# checks whether the value in the "Income" column is greater than or equal to 1730 and les
s than 35196. If the condition is true, the value in the "Teenhome" column is used for the
new column. Otherwise, the value 3 is used for the new column.
data['teens for low income customer'] = np.where((data['Income'] >= 1730) & (data['Incom
e'] < 35196) , data['Teenhome'] , 3)
```

In [76]:

```
# Convert a pandas DataFrame column to a numpy array
array_teens1 = data['teens for low income customer'].values
```

In [77]:

```
# checks whether the value in the "teens for low income customer" column of the pandas Dat
aFrame object named "data" is equal to 0.
df_low_zero_teen = (data['teens for low income customer'] == 0)
data[df_low_zero_teen]
```

Out[77]:

	Income	Kidhome	Teenhome	Recency	MntWines	MntFruits	MntMeatProducts	MntFishPr
3	26646.0	1	0	26	11	4	20	
7	33454.0	1	0	32	76	10	56	
8	30351.0	1	0	19	14	0	24	
10	7500.0	0	0	59	6	16	11	
13	17323.0	0	0	38	3	14	17	
...	...	...	...	...	...	...	...	
2189	7500.0	1	0	7	2	8	11	
2195	24434.0	2	0	9	3	2	8	
2196	11012.0	1	0	82	24	3	26	
2198	26816.0	0	0	50	5	1	6	
2199	34421.0	1	0	81	3	3	7	

417 rows × 43 columns

In [78]:

```
# groups the pandas DataFrame named "data" by the categorical variable "teens for low inco
me customer" using the "groupby" method.
lowIncome_totalTeensAcceptedCmp = data.groupby("teens for low income customer")[['Accepted
Cmp1', 'AcceptedCmp2', 'AcceptedCmp3', 'AcceptedCmp4', 'AcceptedCmp5']].sum()
```

Number of customers who accept the offer in each campaign and their income is low based on the number of teens

In [79]:

```
# Display a pandas DataFrame object showing the total number of accepted campaigns for eac
h category of a categorical variable
lowIncome_totalTeensAcceptedCmp
```

Out[79]:

teens for low income customer	AcceptedCmp1	AcceptedCmp2	AcceptedCmp3	AcceptedCmp4	AcceptedCmp5
0	0	0	36	0	0
1	0	0	7	1	0
2	0	0	0	0	0
3	142	30	120	163	161

How many customers with low income who don't have teens and accepted the offer in each campaign ?

The number of low-income customers who don't have teens and accepted the offer in the first campaign is zero

The number of low-income customers who don't have teens and accepted the offer in the second campaign is zero

The number of low-income customers who don't have teens and accepted the offer in the third campaign is 36

The number of low-income customers who don't have teens and accepted the offer in the fourth campaign is zero

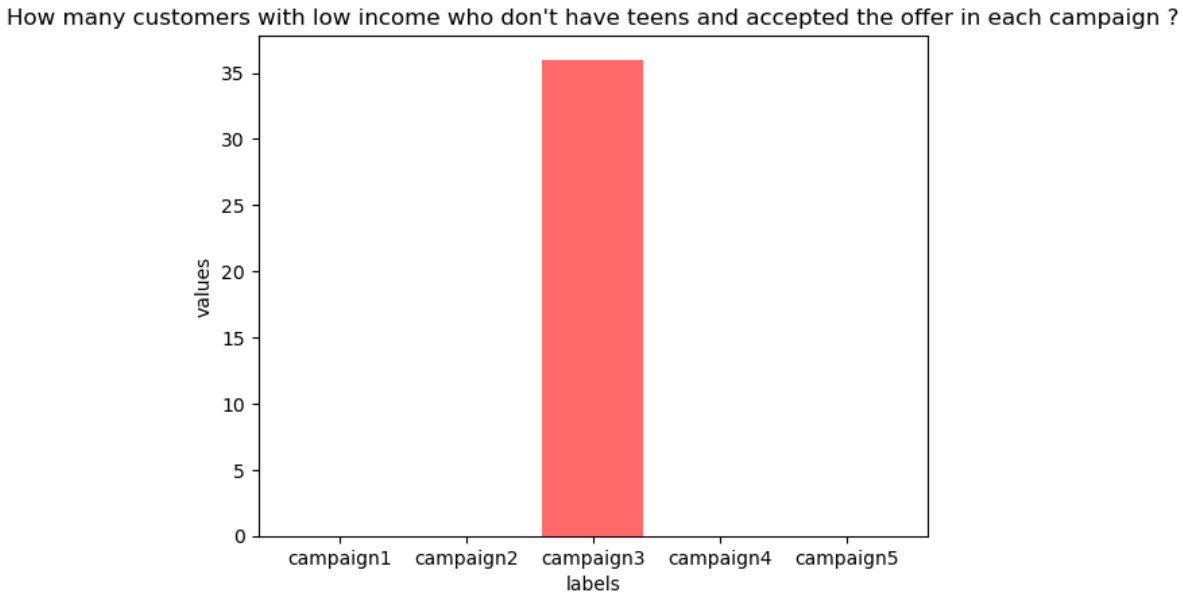
The number of low-income customers who don't have teens and accepted the offer in the fifth campaign is zero

In [80]:

```
# Visualize the number of accepted campaigns for low income customers with zero teens
# Define the values for the bar chart
low_income_customers_campaigns_zero_teens = [0, 0, 36, 0, 0]
# Define the labels for the x-axis
labels = ["campaign1", "campaign2", "campaign3", "campaign4", "campaign5"]
# Set the positions and labels of the x-ticks
plt.xticks(range(len(low_income_customers_campaigns_zero_teens)), labels)
# Add a label for the x-axis
plt.xlabel('labels')
# Add a label for the y-axis
plt.ylabel('values')
# Add a title to the plot
plt.title('How many customers with low income who don\'t have teens and accepted the offer in each campaign ?')
# Create a bar chart with the specified values
plt.bar(range(len(low_income_customers_campaigns_zero_teens)), low_income_customers_campaigns_zero_teens, color="#FF6969")
```

Out[80]:

```
<BarContainer object of 5 artists>
```



In [81]:

```
# checks whether the value in the "teens for low income customer" column of the pandas Dat
aFrame object named "data" is equal to 1.
df_low_one_teen = (data['teens for low income customer'] == 1)
data[df_low_one_teen]
```

Out[81]:

	Income	Kidhome	Teenhome	Recency	MntWines	MntFruits	MntMeatProducts	MntFishPr
9	5648.0	1	1	68	28	0	6	
40	21994.0	0	1	4	9	0	6	
61	32474.0	1	1	0	10	0	1	
79	29440.0	1	1	95	17	8	14	
92	34554.0	0	1	43	41	1	6	
...	...	...	...	...	...	...	...	
2124	34176.0	0	1	9	11	2	7	
2162	8820.0	1	1	52	12	0	13	
2174	32144.0	1	1	76	41	0	10	
2175	14918.0	0	1	52	3	3	3	
2181	5305.0	0	1	12	12	4	7	

131 rows × 43 columns

How many customers with low income who have one teen and accepted the offer in each campaign ?

The number of low-income customers who have one teen and accepted the offer in the first campaign is zero

The number of low-income customers who have one teen and accepted the offer in the second campaign is zero

The number of low-income customers who have one teen and accepted the offer in the third campaign is 7

The number of low-income customers who have one teen and accepted the offer in the fourth campaign is 1

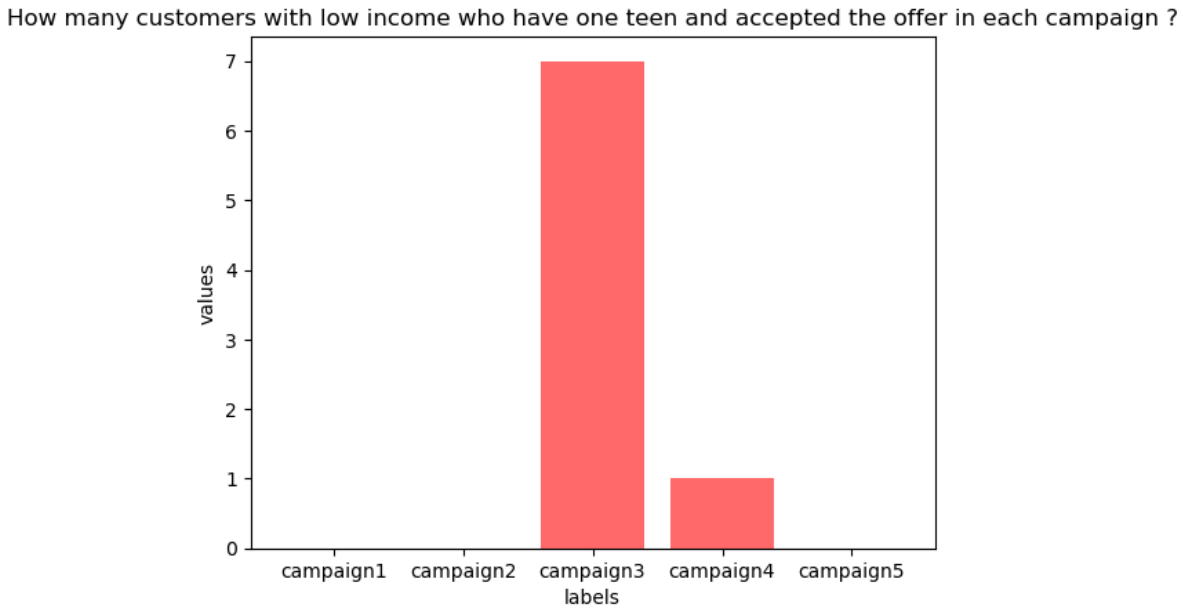
The number of low-income customers who have one teen and accepted the offer in the fifth campaign is zero

In [82]:

```
# Visualize the number of accepted campaigns for low income customers with one teen
# Define the values for the bar chart
low_income_customers_campaigns_one_teen = [0, 0, 7, 1, 0]
# Define the labels for the x-axis
labels = ["campaign1", "campaign2", "campaign3", "campaign4", "campaign5"]
# Set the positions and labels of the x-ticks
plt.xticks(range(len(low_income_customers_campaigns_one_teen)), labels)
# Add a label for the x-axis
plt.xlabel('labels')
# Add a label for the y-axis
plt.ylabel('values')
# Add a title to the plot
plt.title('How many customers with low income who have one teen and accepted the offer in each campaign ?')
# Create a bar chart with the specified values
plt.bar(range(len(low_income_customers_campaigns_one_teen)), low_income_customers_campaigns_one_teen, color="#FF6969")
```

Out[82]:

```
<BarContainer object of 5 artists>
```



In [83]:

```
# checks whether the value in the "teens for low income customer" column of the pandas DataFrame object named "data" is equal to 2.
df_low_two_teen = (data['teens for low income customer'] == 2)
data[df_low_two_teen]
```

Out[83]:

	Income	Kidhome	Teenhome	Recency	MntWines	MntFruits	MntMeatProducts	MntFishPr
1780	7144.0	0	2	92	81	4	33	
1993	30261.0	1	2	75	8	0	5	
2190	33562.0	1	2	33	21	12	12	

3 rows × 43 columns

How many customers with low income who have two teens and accepted the offer in each campaign ?

The number of low-income customers who have two teens and accepted the offer in the first campaign is zero

The number of low-income customers who have two teens and accepted the offer in the second campaign is zero

The number of low-income customers who have two teens and accepted the offer in the third campaign is zero

The number of low-income customers who have two teens and accepted the offer in the fourth campaign is zero

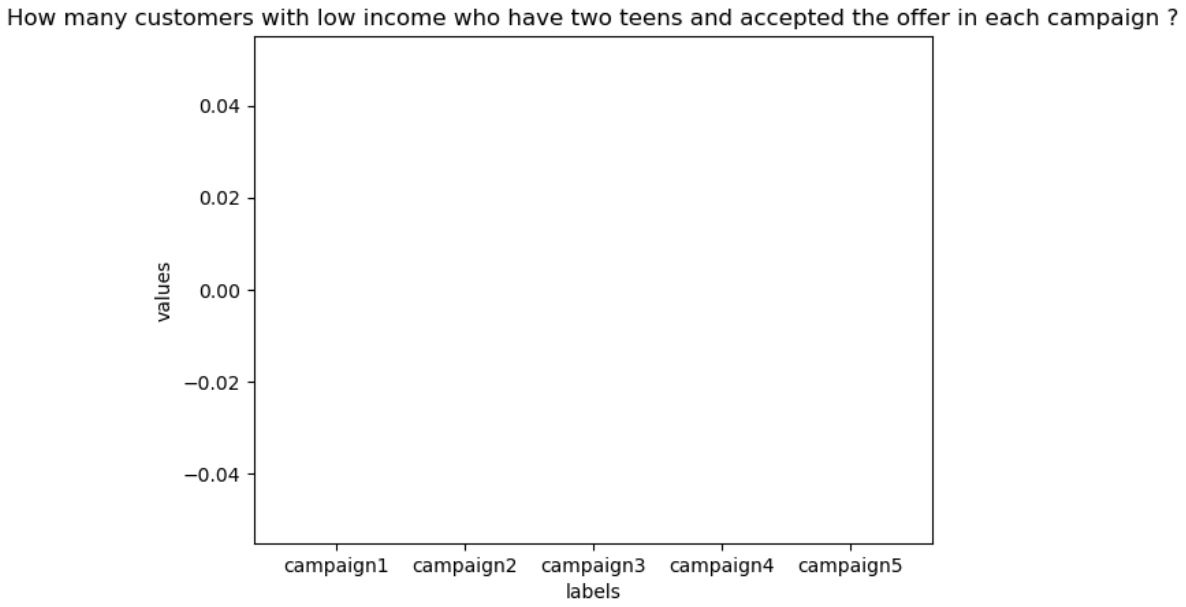
The number of low-income customers who have two teens and accepted the offer in the fifth campaign is zero

In [84]:

```
# Visualize the number of accepted campaigns for low income customers with two teens
# Define the values for the bar chart
low_income_customers_campaigns_two_teens = [0, 0, 0, 0, 0]
# Define the labels for the x-axis
labels = ["campaign1", "campaign2", "campaign3", "campaign4", "campaign5"]
# Set the positions and labels of the x-ticks
plt.xticks(range(len(low_income_customers_campaigns_two_teens)), labels)
# Add a label for the x-axis
plt.xlabel('labels')
# Add a label for the y-axis
plt.ylabel('values')
# Add a title to the plot
plt.title('How many customers with low income who have two teens and accepted the offer in
each campaign ?')
# Create a bar chart with the specified values
plt.bar(range(len(low_income_customers_campaigns_two_teens)), low_income_customers_campaig
ns_two_teens, color="#FF6969")
```

Out[84]:

```
<BarContainer object of 5 artists>
```



In [85]:

```
# Count the number of low income customers with zero teens
low_ZeroTeens = 0
for count_low0teens in array_teens1:
    if count_low0teens == 0:
        low_ZeroTeens+=1
    else:
        continue
```

In [86]:

```
# Print the number of low income customers with zero teens
print(low_ZeroTeens)
```

```
417
```

In [87]:

```
# Count the number of low income customers with one teen
low_oneTeen = 0
for count_low1teens in array_teens1:
    if count_low1teens == 1:
        low_oneTeen+=1
    else:
        continue
```

In [88]:

```
# Print the number of low income customers with one teen
print(low_oneTeen)
```

```
131
```

In [89]:

```
# Count the number of low income customers with two teens
low_twoTeen = 0
for count_low2teens in array_teens1:
    if count_low2teens == 2:
        low_twoTeen+=1
    else:
        continue
```

In [90]:

```
# Print the number of low income customers with one teen
print(low_twoTeen)
```

```
3
```

Number of teens for each customer based on their low income

76% from Customers whose income is low do not have teens in their home

24% from Customers whose income is low have one teen in their home

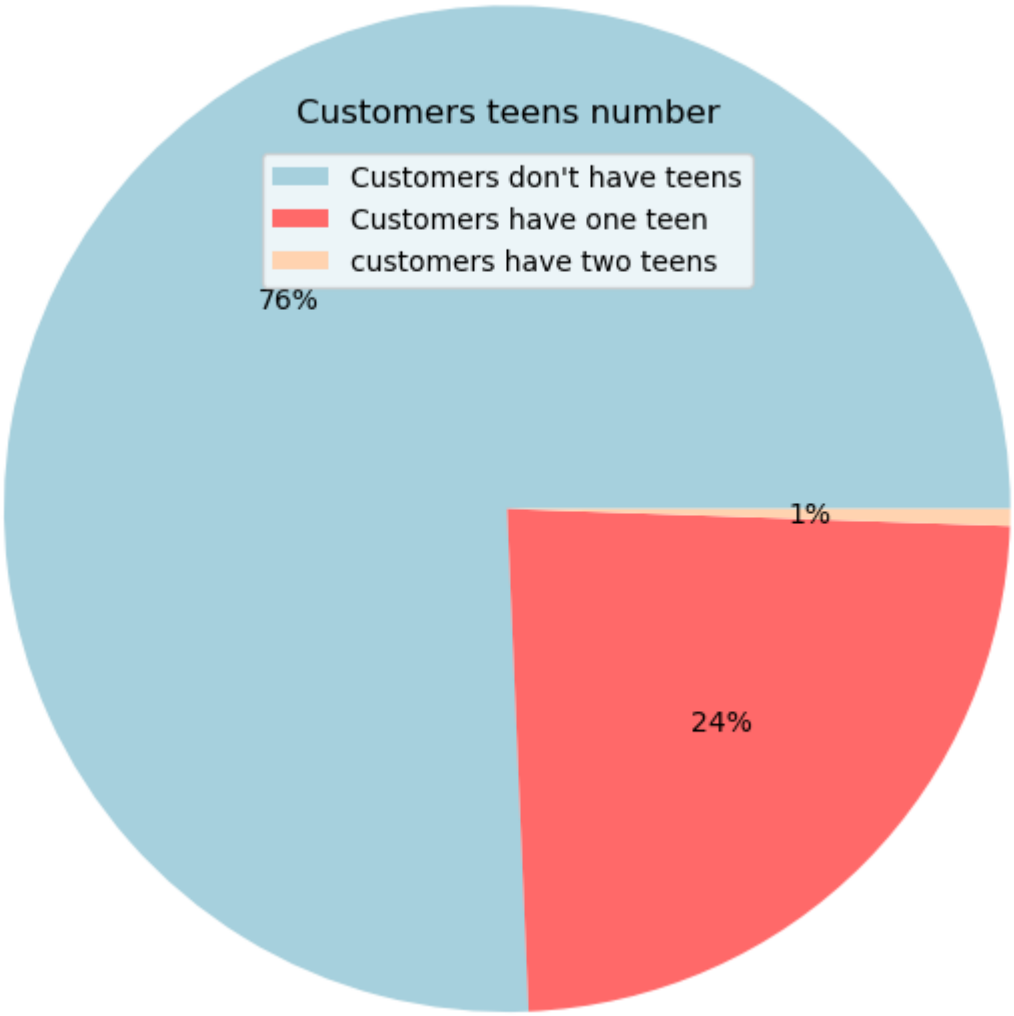
1% from Customers whose income is low have two teens in their home

In [91]:

```
# Visualize the number of low income customers with different numbers of teens
# Define the values for the pie chart
low_income_teens =[417, 131, 3]
# Create a pie chart with the specified values
plt.pie(low_income_teens,  radius=1.7, colors=["#A6D0DD","#FF6969", "#FFD3B0"],
        rotatelabels=False, autopct= "%1.0f%%")
# Add a title to the plot
plt.title('Customers teens number')
# Add a legend to the plot with the specified labels
plt.legend(labels=["Customers don't have teens", "Customers have one teen", "customers hav
e two teens"])
```

Out[91]:

```
<matplotlib.legend.Legend at 0x78000661f760>
```



In [92]:

```
# checks whether the value in the "Income" column is greater than or equal to 35196 and le
ss than 68281. If the condition is true, the value in the "Teenhome" column is used for th
e new column. Otherwise, the value 3 is used for the new column.
data['teens for average income customer'] = np.where((data['Income'] >= 35196) & (data['In
come'] < 68281) , data['Teenhome'] , 3)
```

In [93]:

```
# Convert a pandas DataFrame column to a numpy array
array_teens2 = data['teens for average income customer'].values
```

In [94]:

```
# checks whether the value in the "teens for average income customer" column of the pandas DataFrame named "data" is equal to 0.
df_average_zero_teen = (data['teens for average income customer'] == 0)
data[df_average_zero_teen]
```

Out[94]:

	Income	Kidhome	Teenhome	Recency	MntWines	MntFruits	MntMeatProducts	MntFishPr
0	58138.0	0	0	58	635	88	546	
4	58293.0	1	0	94	173	43	118	
11	63033.0	0	0	82	194	61	480	
16	37760.0	0	0	20	84	5	38	
19	37040.0	0	0	41	86	2	73	
...	...	...	...	...	...	...	...	
2155	65487.0	0	0	48	240	67	500	
2163	43322.0	0	0	25	56	7	48	
2186	40101.0	1	0	73	171	3	129	
2197	44802.0	0	0	71	853	10	143	
2202	56981.0	0	0	91	908	48	217	

317 rows × 44 columns

In [95]:

```
# groups the pandas DataFrame named "data" by the categorical variable "teens for average income customer" using the "groupby" method.
averageIncome_totalTeensAcceptedCmp = data.groupby("teens for average income customer")
[['AcceptedCmp1', 'AcceptedCmp2', 'AcceptedCmp3', 'AcceptedCmp4', 'AcceptedCmp5']].sum()
```

Number of customers who accept the offer in each campaign and their income is average based on the number of teens

In [96]:

```
# Display a pandas DataFrame object showing the total number of accepted campaigns for eac
h category of a categorical variable
averageIncome_totalTeensAcceptedCmp
```

Out[96]:

teens for average income customer	AcceptedCmp1	AcceptedCmp2	AcceptedCmp3	AcceptedCmp4	AcceptedCmp5
0	5	5	29	16	5
1	15	8	44	67	3
2	1	1	4	3	1
3	121	16	86	78	152

How many customers with average income who don't have teens and accepted the offer in each campaign ?

The number of average-income customers who don't have teens and accepted the offer in the first campaign is 5

The number of average-income customers who don't have teens and accepted the offer in the second campaign is 5

The number of average-income customers who don't have teens and accepted the offer in the third campaign is 29

The number of average-income customers who don't have teens and accepted the offer in the fourth campaign is 16

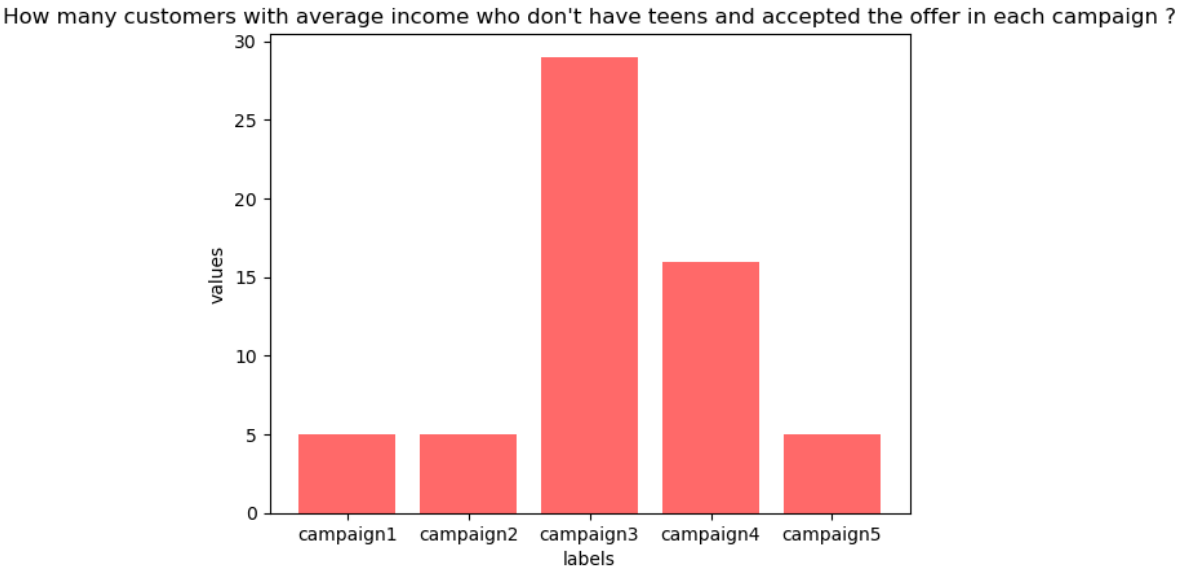
The number of average-income customers who don't have teens and accepted the offer in the fifth campaign is 5

In [97]:

```
# Visualize the number of accepted campaigns for average income customers with zero teens
# Define the values for the bar chart
average_income_customers_campaigns_zero_teens = [5, 5, 29, 16, 5]
# Define the labels for the x-axis
labels = ["campaign1", "campaign2", "campaign3", "campaign4", "campaign5"]
# Set the positions and labels of the x-ticks
plt.xticks(range(len(average_income_customers_campaigns_zero_teens)), labels)
# Add a label for the x-axis
plt.xlabel('labels')
# Add a label for the y-axis
plt.ylabel('values')
# Add a title to the plot
plt.title('How many customers with average income who don\'t have teens and accepted the o
ffer in each campaign ?')
# Create a bar chart with the specified values
plt.bar(range(len(average_income_customers_campaigns_zero_teens)), average_income_customer
s_campaigns_zero_teens, color="#FF6969")
```

Out[97]:

```
<BarContainer object of 5 artists>
```



In [98]:

```
# checks whether the value in the "teens for average income customer" column of the pandas DataFrame named "data" is equal to 1.
df_average_one_teen = (data['teens for average income customer'] == 1)
data[df_average_one_teen]
```

Out[98]:

	Income	Kidhome	Teenhome	Recency	MntWines	MntFruits	MntMeatProducts	MntFishPr
1	46344.0	1	1	38	11	1	6	
5	62513.0	0	1	16	520	42	98	
6	55635.0	0	1	34	235	65	164	
12	59354.0	1	1	53	233	2	53	
15	41850.0	1	1	51	53	5	19	
...	...	...	...	...	...	...	...	
2193	63777.0	1	1	87	457	5	106	
2194	57967.0	0	1	39	229	7	137	
2200	61223.0	0	1	46	709	43	182	
2201	64014.0	2	1	56	406	0	30	
2204	52869.0	1	1	40	84	3	61	

745 rows × 44 columns

How many customers with average income who have one teen and accepted the offer in each campaign ?

The number of average-income customers who have one teen and accepted the offer in the first campaign is 15

The number of average-income customers who have one teen and accepted the offer in the second campaign is 8

The number of average-income customers who have one teen and accepted the offer in the third campaign is 44

The number of average-income customers who have one teen and accepted the offer in the fourth campaign is 67

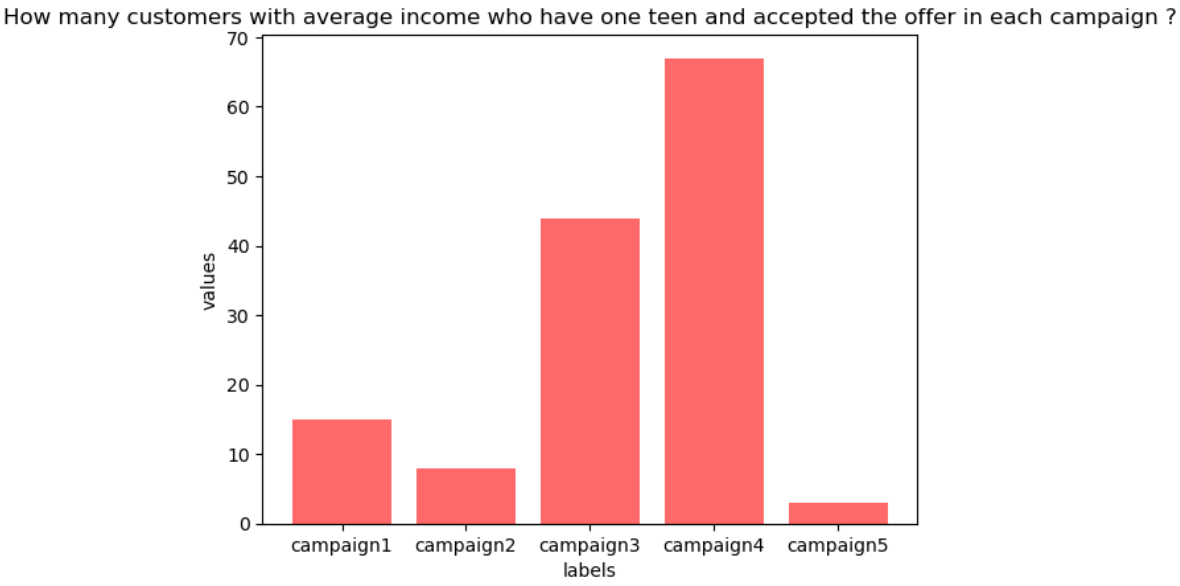
The number of average-income customers who have one teen and accepted the offer in the fifth campaign is 3

In [99]:

```
# Visualize the number of accepted campaigns for average income customers with one teen
# Define the values for the bar chart
average_income_customers_campaigns_one_teen = [15, 8, 44, 67, 3]
# Define the labels for the x-axis
labels = ["campaign1", "campaign2", "campaign3", "campaign4", "campaign5"]
# Set the positions and labels of the x-ticks
plt.xticks(range(len(average_income_customers_campaigns_one_teen)), labels)
# Add a label for the x-axis
plt.xlabel('labels')
# Add a label for the y-axis
plt.ylabel('values')
# Add a title to the plot
plt.title('How many customers with average income who have one teen and accepted the offer in each campaign ?')
# Create a bar chart with the specified values
plt.bar(range(len(average_income_customers_campaigns_one_teen)), average_income_customers_campaigns_one_teen, color="#FF6969")
```

Out[99]:

```
<BarContainer object of 5 artists>
```



In [100]:

```
# checks whether the value in the "teens for average income customer" column of the pandas
DataFrame named "data" is equal to 2.
df_average_two_teen = (data['teens for average income customer'] == 2)
data[df_average_two_teen]
```

Out[100]:

	Income	Kidhome	Teenhome	Recency	MntWines	MntFruits	MntMeatProducts	MntFishPr
31	46610.0	0	2	8	96	12	96	
98	52413.0	0	2	56	295	106	271	
134	59809.0	0	2	36	598	16	141	
139	59354.0	0	2	59	295	21	78	
146	60199.0	1	2	49	8	1	7	
319	62204.0	0	2	38	317	46	247	
342	46681.0	0	2	52	269	15	69	
387	55521.0	1	2	11	416	0	26	
392	50437.0	0	2	28	370	9	92	
408	48686.0	1	2	8	10	0	7	
438	38988.0	1	2	90	164	24	103	
451	53790.0	0	2	86	335	42	127	
495	48920.0	0	2	93	238	17	68	
586	61467.0	0	2	69	410	16	114	
640	46734.0	1	2	86	100	1	39	
652	59247.0	0	2	87	327	9	122	
675	61923.0	0	2	94	92	4	18	
713	45072.0	1	2	74	144	2	99	
722	48767.0	1	2	79	28	1	21	
785	49681.0	0	2	66	411	0	26	
807	37859.0	1	2	75	22	1	8	
812	56575.0	0	2	42	421	5	90	
884	58917.0	1	2	10	151	7	89	
1002	64504.0	1	2	81	986	36	168	
1041	65196.0	0	2	34	743	19	181	
1147	35322.0	1	2	34	28	9	37	
1169	45894.0	0	2	15	27	2	7	
1188	40451.0	0	2	54	35	0	4	
1231	67433.0	0	2	51	615	28	259	
1257	40800.0	1	2	77	24	0	27	
1355	56243.0	1	2	26	347	0	35	
1370	63915.0	0	2	2	622	7	115	
1518	51411.0	1	2	81	14	0	3	

	Income	Kidhome	Teenhome	Recency	MntWines	MntFruits	MntMeatProducts	MntFishPr
1536	49681.0	0	2	66	411	0	26	
1609	46681.0	0	2	52	269	15	69	
1616	64140.0	0	2	71	1459	0	61	
1640	63246.0	0	2	60	593	30	91	
1772	63404.0	0	2	97	734	26	70	
1792	50387.0	0	2	91	369	9	87	
2114	41275.0	1	2	33	24	4	22	

40 rows × 44 columns

How many customers with average income who have two teens and accepted the offer in each campaign ?

The number of average-income customers who have two teens and accepted the offer in the first campaign is 1

The number of average-income customers who have two teens and accepted the offer in the second campaign is 1

The number of average-income customers who have two teens and accepted the offer in the third campaign is 4

The number of average-income customers who have two teens and accepted the offer in the fourth campaign is 3

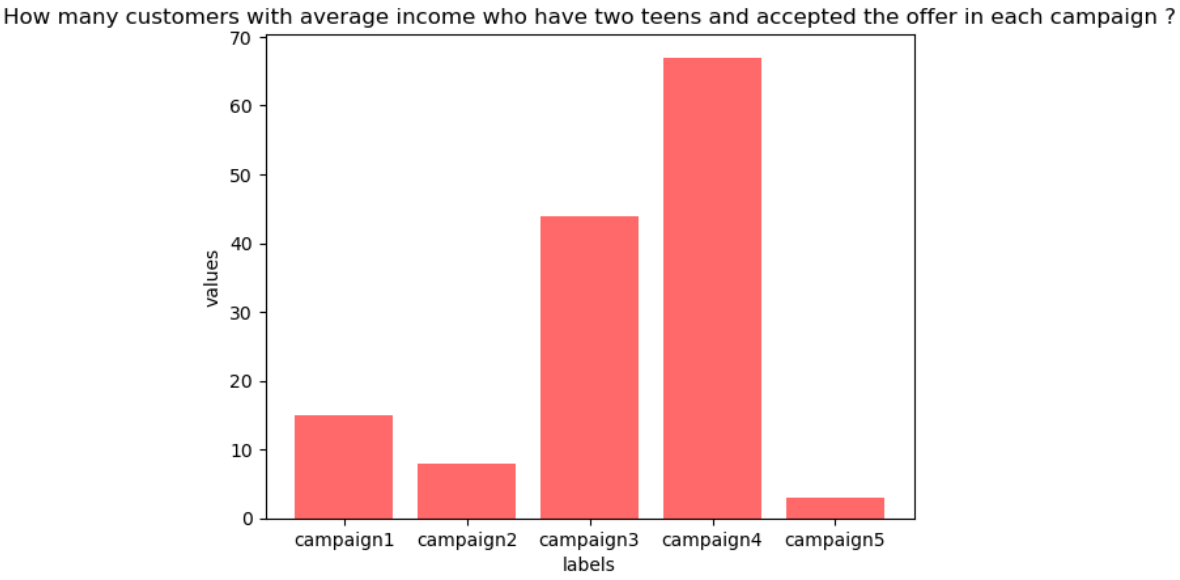
The number of average-income customers who have two teens and accepted the offer in the fifth campaign is 1

In [101]:

```
# Visualize the number of accepted campaigns for low income customers with two teens
# Define the values for the bar chart
average_income_customers_campaigns_two_teens = [15, 8, 44, 67, 3]
# Define the labels for the x-axis
labels = ["campaign1", "campaign2", "campaign3", "campaign4", "campaign5"]
# Set the positions and labels of the x-ticks
plt.xticks(range(len(average_income_customers_campaigns_two_teens)), labels)
# Add a label for the x-axis
plt.xlabel('labels')
# Add a label for the y-axis
plt.ylabel('values')
# Add a title to the plot
plt.title('How many customers with average income who have two teens and accepted the offe
r in each campaign ?')
# Create a bar chart with the specified values
plt.bar(range(len(average_income_customers_campaigns_two_teens)), average_income_customers
_campaigns_two_teens, color="#FF6969")
```

Out[101]:

```
<BarContainer object of 5 artists>
```



In [102]:

```
# Count the number of average income customers with zero teens
average_ZeroTeens = 0
for count_average0teens in array_teens2:
    if count_average0teens == 0:
        average_ZeroTeens+=1
    else:
        continue
```

In [103]:

```
# display number of average income customers with zero teens
print(average_ZeroTeens)
```

317

In [104]:

```
# Count the number of average income customers with one teen
average_oneTeens = 0
for count_average1teens in array_teens2:
    if count_average1teens == 1:
        average_oneTeens+=1
    else:
        continue
```

In [105]:

```
# display number of average income customers with one teen
print(average_oneTeens)
```

745

In [106]:

```
# Count the number of average income customers with two teens
average_twoTeens = 0
for count_average2teens in array_teens2:
    if count_average2teens == 2:
        average_twoTeens+=1
    else:
        continue
```

In [107]:

```
# display number of average income customers with two teens
print(average_twoTeens)
```

40

Number of teens for each customer based on their average income

29% from Customers whose income is average do not have teens in their home

68% from Customers whose income is average have one teen in their home

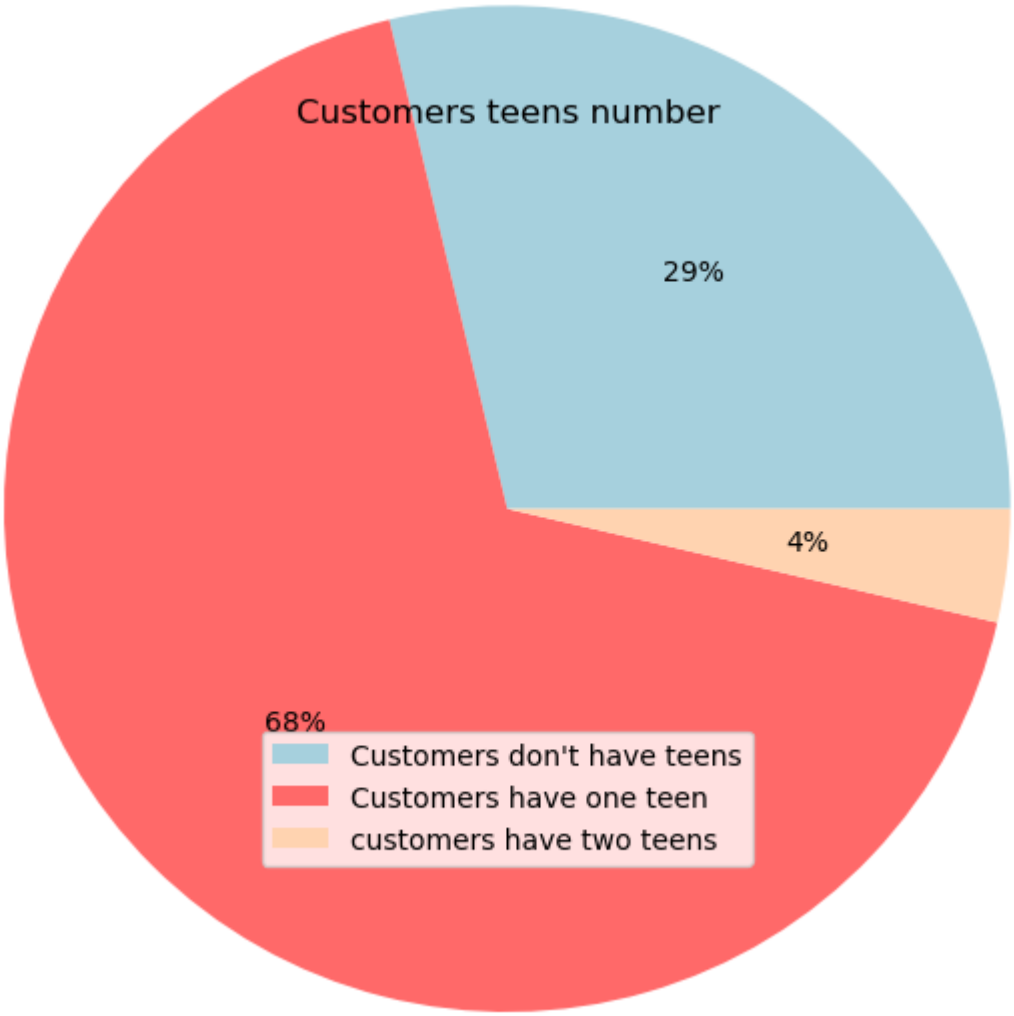
3% from Customers whose income is average have two teens in their home

In [108]:

```
# Visualize the number of average income customers with different numbers of teens
# Define the values for the pie chart
avarage_income_teens =[317, 745, 40]
# Create a pie chart with the specified values
plt.pie(avarage_income_teens,  radius=1.7, colors=["#A6D0DD","#FF6969", "#FFD3B0"],
        rotatelabels=False, autopct= "%1.0f%%")
# Add a title to the plot
plt.title('Customers teens number')
# Add a legend to the plot with the specified labels
plt.legend(labels=["Customers don't have teens", "Customers have one teen", "customers hav
e two teens"])
```

Out[108]:

```
<matplotlib.legend.Legend at 0x7800063bbdc0>
```



In [109]:

```
# checks whether the value in the "Income" column is greater than or equal to 68281 and le
ss than or equal to 113734. If the condition is true, the value in the "Teenhome" column i
s used for the new column. Otherwise, the value 3 is used for the new column.
data['teens for high income customer'] = np.where((data['Income'] >= 68281) & (data['Incom
e'] <= 113734) , data['Teenhome'] , 3)
```

In [110]:

```
# Convert a pandas DataFrame column to a numpy array
array_teen3 = data['teens for high income customer'].values
```

In [111]:

```
# checks whether the value in the "teens for high income customer" column of the pandas Da
taFrame named "data" is equal to 0.
df_high_zero_teen = (data['teens for high income customer'] == 0)
data[df_high_zero_teen]
```

Out[111]:

	Income	Kidhome	Teenhome	Recency	MntWines	MntFruits	MntMeatProducts	MntFishPr
2	71613.0	0	0	26	426	49	127	
14	82800.0	0	0	23	1006	22	115	
27	84618.0	0	0	96	684	100	801	
32	68657.0	0	0	4	482	34	471	
42	79941.0	0	0	72	123	164	266	
...	...	...	...	...	...	...	...	
2160	82347.0	0	0	38	556	54	845	
2178	88325.0	0	0	42	519	71	860	
2180	80617.0	0	0	42	594	51	631	
2184	82032.0	0	0	54	332	194	377	
2188	75777.0	0	0	12	712	26	538	

405 rows × 45 columns

In [112]:

```
# groups the pandas DataFrame named "data" by the categorical variable "teens for high inc
ome customer" using the "groupby" method.
highIncome_totalTeensAcceptedCmp = data.groupby("teens for high income customer")[['Accept
edCmp1', 'AcceptedCmp2', 'AcceptedCmp3', 'AcceptedCmp4', 'AcceptedCmp5']].sum()
```

Number of customers who accept the offer in each campaign and their income is high based on the number of teens

In [113]:

```
# Display a pandas DataFrame object showing the total number of accepted campaigns for eac
h category of a categorical variable
highIncome_totalTeensAcceptedCmp
```

Out[113]:

teens for high income customer	AcceptedCmp1	AcceptedCmp2	AcceptedCmp3	AcceptedCmp4	AcceptedCmp5
0	109	13	33	58	137
1	12	3	10	17	13
2	0	0	0	2	2
3	21	14	120	87	9

How many customers with high income who don't have teens and accepted the offer in each campaign ?

The number of high-income customers who don't have teens and accepted the offer in the first campaign is 109

The number of high-income customers who don't have teens and accepted the offer in the second campaign is 13

The number of high-income customers who don't have teens and accepted the offer in the third campaign is 33

The number of high-income customers who don't have teens and accepted the offer in the fourth campaign is 58

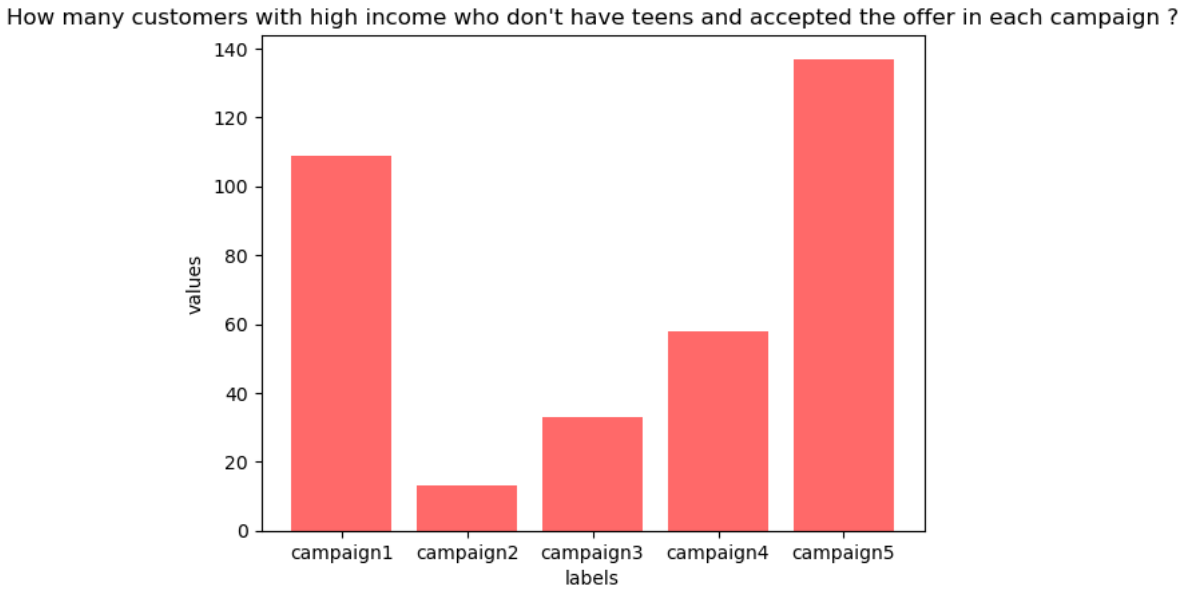
The number of high-income customers who don't have teens and accepted the offer in the fifth campaign is 137

In [114]:

```
# Visualize the number of accepted campaigns for high income customers with zero teens
# Define the values for the bar chart
high_income_customers_campaigns_zero_teens = [109, 13, 33, 58, 137]
# Define the labels for the x-axis
labels = ["campaign1", "campaign2", "campaign3", "campaign4", "campaign5"]
# Set the positions and labels of the x-ticks
plt.xticks(range(len(high_income_customers_campaigns_zero_teens)), labels)
# Add a label for the x-axis
plt.xlabel('labels')
# Add a label for the y-axis
plt.ylabel('values')
# Add a title to the plot
plt.title('How many customers with high income who don\'t have teens and accepted the offe
r in each campaign ?')
# Create a bar chart with the specified values
plt.bar(range(len(high_income_customers_campaigns_zero_teens)), high_income_customers_camp
aigns_zero_teens, color="#FF6969")
```

Out[114]:

```
<BarContainer object of 5 artists>
```



In [115]:

```
# checks whether the value in the "teens for high income customer" column of the pandas DataFrame named "data" is equal to 1.
df_high_one_teen = (data['teens for high income customer'] == 1)
data[df_high_one_teen]
```

Out[115]:

	Income	Kidhome	Teenhome	Recency	MntWines	MntFruits	MntMeatProducts	MntFishPr
17	76995.0	0	1	91	1012	80	498	
38	80011.0	0	1	3	421	76	536	
45	72550.0	1	1	39	826	50	317	
62	88194.0	0	1	19	688	14	309	
63	69096.0	0	1	4	247	49	159	
...	...	...	...	...	...	...	...	
2140	71965.0	0	1	21	572	19	286	
2153	76234.0	0	1	21	519	50	167	
2161	73803.0	0	1	61	833	80	363	
2170	73807.0	0	1	88	366	124	156	
2203	69245.0	0	1	8	428	30	214	

139 rows × 45 columns

How many customers with high income who have one teen and accepted the offer in each campaign ?

The number of high-income customers who have one teen and accepted the offer in the first campaign is 12

The number of high-income customers who have one teen and accepted the offer in the second campaign is 3

The number of high-income customers who have one teen and accepted the offer in the third campaign is 10

The number of high-income customers who have one teen and accepted the offer in the fourth campaign is 17

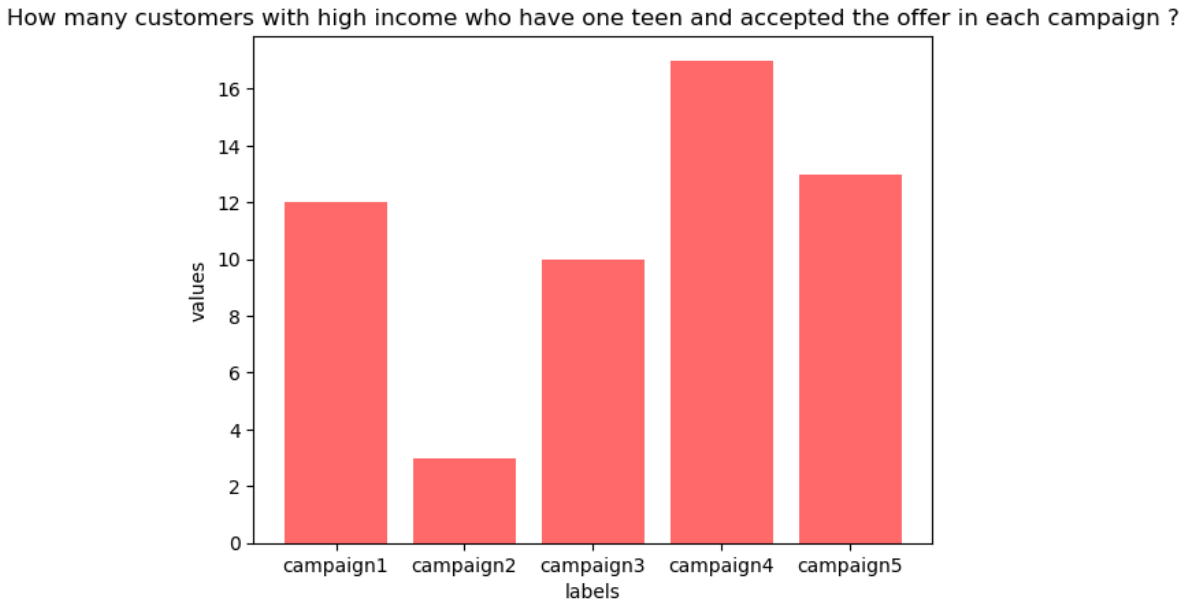
The number of high-income customers who have one teen and accepted the offer in the fifth campaign is 13

In [116]:

```
# Visualize the number of accepted campaigns for high income customers with one teen
# Define the values for the bar chart
high_income_customers_campaigns_one_teen = [12, 3, 10, 17, 13]
# Define the labels for the x-axis
labels = ["campaign1", "campaign2", "campaign3", "campaign4", "campaign5"]
# Set the positions and labels of the x-ticks
plt.xticks(range(len(high_income_customers_campaigns_one_teen)), labels)
# Add a label for the x-axis
plt.xlabel('labels')
# Add a label for the y-axis
plt.ylabel('values')
# Add a title to the plot
plt.title('How many customers with high income who have one teen and accepted the offer in each campaign ?')
# Create a bar chart with the specified values
plt.bar(range(len(high_income_customers_campaigns_one_teen)), high_income_customers_campaigns_one_teen, color="#FF6969")
```

Out[116]:

```
<BarContainer object of 5 artists>
```



In [117]:

```
# checks whether the value in the "teens for high income customer" column of the pandas Da
taFrame named "data" is equal to 2.
df_high_two_teen = (data['teens for high income customer'] == 2)
data[df_high_two_teen]
```

Out[117]:

	Income	Kidhome	Teenhome	Recency	MntWines	MntFruits	MntMeatProducts	MntFishPr
64	74854.0	1	2	90	856	59	487	
209	77622.0	0	2	3	520	7	154	
247	69674.0	0	2	46	554	41	215	
806	93404.0	1	2	97	1279	15	287	
1448	94871.0	0	2	99	169	24	553	
1544	75283.0	1	2	26	733	9	180	
1673	73705.0	0	2	86	612	91	520	
2015	83273.0	1	2	98	433	89	650	

8 rows × 45 columns

How many customers with high income who have two teens and accepted the offer in each campaign ?

The number of high-income customers who have two teens and accepted the offer in the first campaign is zero

The number of high-income customers who have two teens and accepted the offer in the second campaign is zero

The number of high-income customers who have two teens and accepted the offer in the third campaign is zero

The number of high-income customers who have two teens and accepted the offer in the fourth campaign is 2

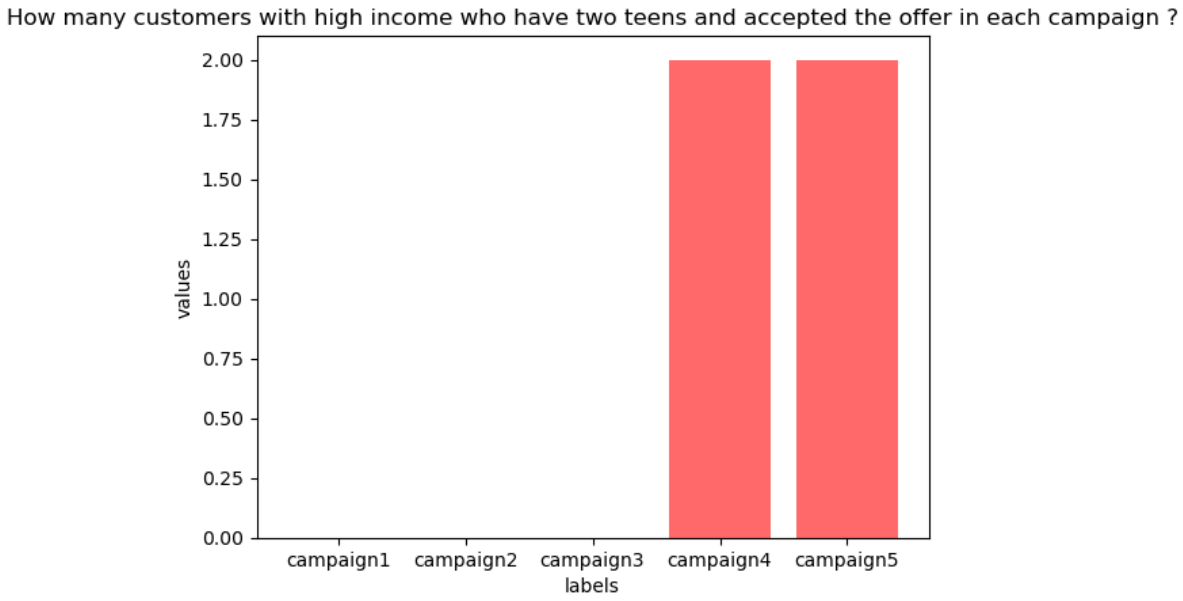
The number of high-income customers who have two teens and accepted the offer in the fifth campaign is 2

In [118]:

```
# Visualize the number of accepted campaigns for high income customers with two teens
# Define the values for the bar chart
high_income_customers_campaigns_two_teens = [0, 0, 0, 2, 2]
# Define the labels for the x-axis
labels = ["campaign1", "campaign2", "campaign3", "campaign4", "campaign5"]
# Set the positions and labels of the x-ticks
plt.xticks(range(len(high_income_customers_campaigns_two_teens)), labels)
# Add a label for the x-axis
plt.xlabel('labels')
# Add a label for the y-axis
plt.ylabel('values')
# Add a title to the plot
plt.title('How many customers with high income who have two teens and accepted the offer i
n each campaign ?')
# Create a bar chart with the specified values
plt.bar(range(len(high_income_customers_campaigns_two_teens)), high_income_customers_campa
igns_two_teens, color="#FF6969")
```

Out[118]:

```
<BarContainer object of 5 artists>
```



In [119]:

```
# Count the number of high income customers with zero teens
high_ZeroTeens = 0
for count_high0teens in array_teen3:
    if count_high0teens == 0:
        high_ZeroTeens+=1
    else:
        continue
```

In [120]:

```
# display the number of high income customers with zero teens
print(high_ZeroTeens)
```

405

In [121]:

```
# Count the number of high income customers with one teen
high_oneTeens = 0
for count_high1teens in array_teen3:
    if count_high1teens == 1:
        high_oneTeens+=1
    else:
        continue
```

In [122]:

```
# display the number of high income customers with one teen
print(high_oneTeens)
```

139

In [123]:

```
# Count the number of high income customers with two teens
high_twoTeens = 0
for count_high2teens in array_teen3:
    if count_high2teens == 2:
        high_twoTeens+=1
    else:
        continue
```

In [124]:

```
# display the number of high income customers with two teens
print(high_twoTeens)
```

8

Number of teens for each customer based on their high income

73% from Customers whose income is high do not have teens in their home

25% from Customers whose income is high have one teen in their home

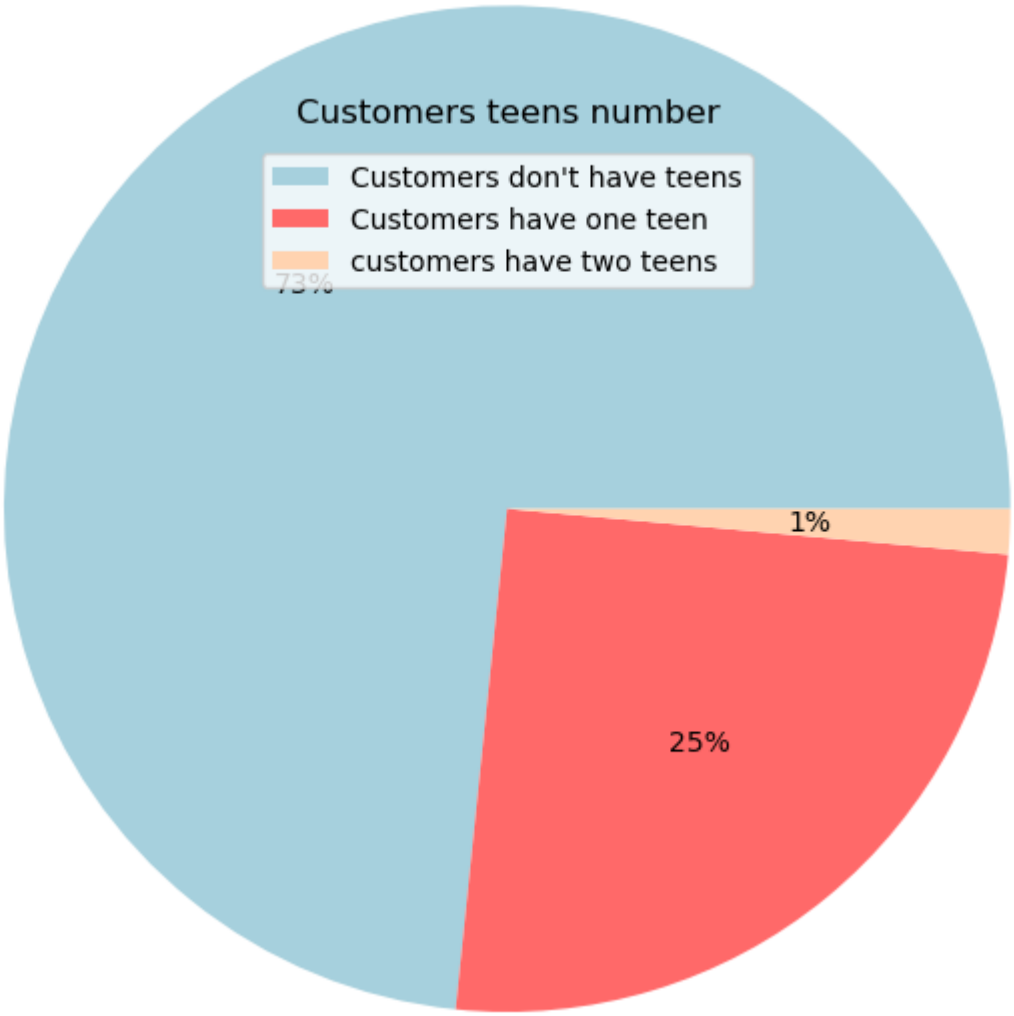
2% from Customers whose income is high have two teens in their home

In [125]:

```
# Visualize the number of high income customers with different numbers of teens
# Define the values for the pie chart
high_income_teens =[405, 139, 8]
# Create a pie chart with the specified values
plt.pie(high_income_teens,  radius=1.7, colors=["#A6D0DD","#FF6969", "#FFD3B0"],
        rotatelabels=False, autopct= "%1.0f%%")
# Add a title to the plot
plt.title('Customers teens number')
# Add a legend to the plot with the specified labels
plt.legend(labels=["Customers don't have teens", "Customers have one teen", "customers hav
e two teens"])
```

Out[125]:

```
<matplotlib.legend.Legend at 0x7800061bce20>
```



In [126]:

```
# calculates the sum of the values in the "MntFishProducts" column
data['MntFishProducts'].sum()
```

Out[126]:

```
83253
```

In [127]:

```
# calculates the sum of the values in the "MntFishProducts" column of the pandas DataFrame object named "data" for a specific subset of data. The subset is defined by the condition that the value in the "Income" column is greater than or equal to 1730 and less than 35196.
fishProduct_lowIncome = data.loc[(data['Income'] >= 1730) & (data['Income'] < 35196), 'MntFishProducts'].sum()
```

In [128]:

```
fishProduct_lowIncome
```

Out[128]:

```
4713
```

In [129]:

```
# calculates the sum of the values in the "MntFishProducts" column of the pandas DataFrame object named "data" for a specific subset of data. The subset is defined by the condition that the value in the "Income" column is greater than or equal to 35196 and less than 635196.
fishProduct_averageIncome = data.loc[(data['Income'] >= 35196) & (data['Income'] < 635196), 'MntFishProducts'].sum()
```

In [130]:

```
fishProduct_averageIncome
```

Out[130]:

```
78540
```

In [131]:

```
# calculates the sum of the values in the "MntFishProducts" column of the pandas DataFrame object named "data" for a specific subset of data. The subset is defined by the condition that the value in the "Income" column is greater than or equal to 635196 and less than 113735.
fishProduct_highIncome = data.loc[(data['Income'] >= 635196) & (data['Income'] <= 113734), 'MntFishProducts'].sum()
```

In [132]:

```
fishProduct_highIncome
```

Out[132]:

```
0
```

In [133]:

```
# amount spent on fish products in the last 2 years for customers
4713/83253
```

Out[133]:

```
0.056610572591978665
```

In [134]:

```
78540/83253
```

Out[134]:

```
0.9433894274080213
```

Plot for company\'s income comes from fish products based on customers-income

6% from the company's income comes from fish products by low-income customers

94% from the company's income comes from fish products by average-income customers

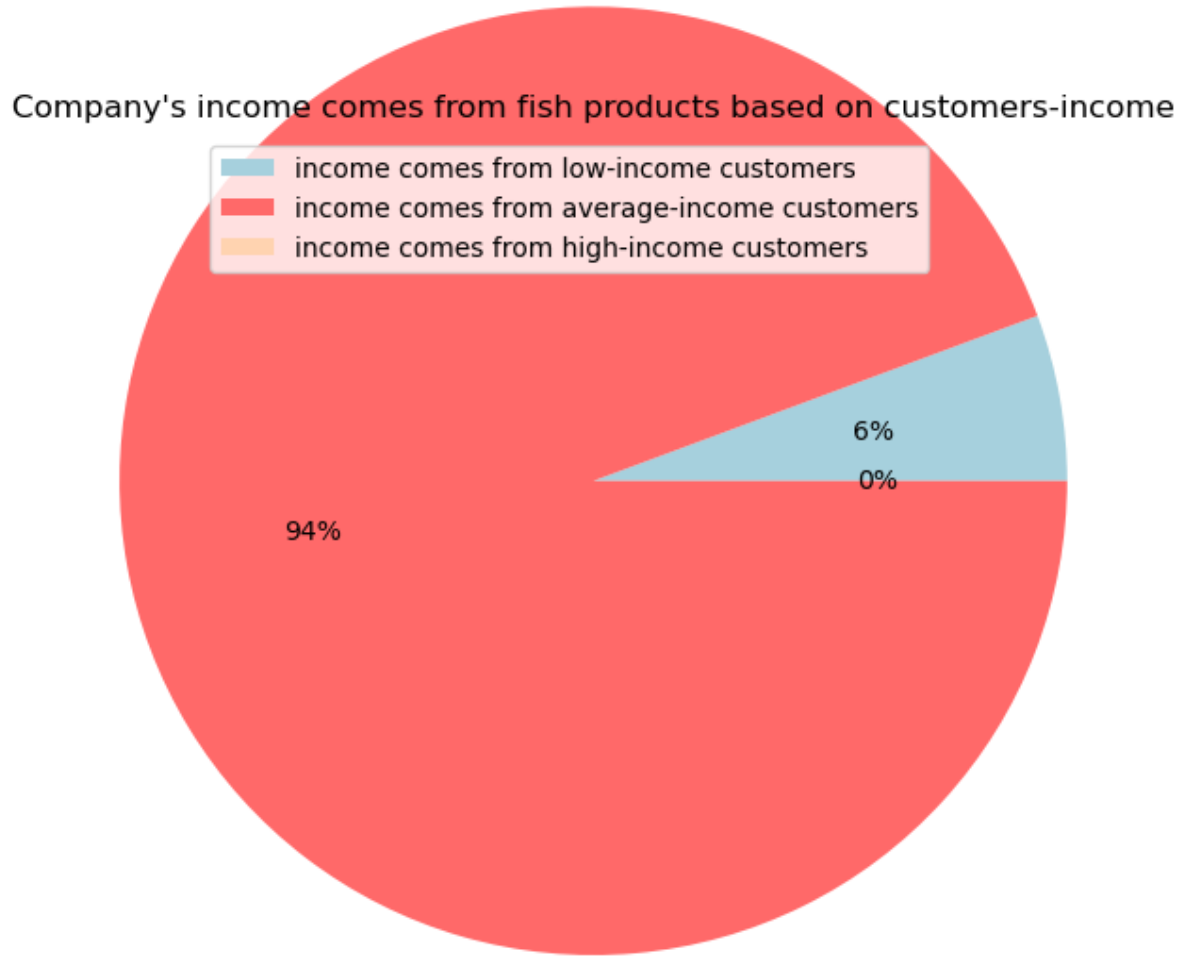
0% from the company's income comes from fish products by high-income customers

In [135]:

```
# Visualize the percentage of company's income comes from fish products based on customers
-income
# Define the values for the pie chart
fishProducts=[0.056610572591978665, 0.9433894274080213, 0]
# Create a pie chart with the specified values
plt.pie(fishProducts,  radius=1.7, colors=["#A6D0DD","#FF6969", "#FFD3B0"],
        rotatelabels=False, autopct= "%1.0f%%")
# Add a title to the plot
plt.title('Company\'s income comes from fish products based on customers-income')
# Add a legend to the plot with the specified labels
plt.legend(labels=["income comes from low-income customers", "income comes from average-in
come customers", "income comes from high-income customers"])
```

Out[135]:

```
<matplotlib.legend.Legend at 0x780006437070>
```



In [136]:

```
# calculates the sum of the values in the "MntMeatProducts" column
data['MntMeatProducts'].sum()
```

Out[136]:

```
364513
```

In [137]:

```
# calculates the sum of the values in the "MntMeatProducts" column of the pandas DataFrame object named "data" for a specific subset of data. The subset is defined by the condition that the value in the "Income" column is greater than or equal to 1730 and less than 35196.
meatProduct_lowIncome = data.loc[(data['Income'] >= 1730) & (data['Income'] < 35196), 'MntMeatProducts'].sum()
```

In [138]:

```
meatProduct_lowIncome
```

Out[138]:

```
12263
```

In [139]:

```
# calculates the sum of the values in the "MntMeatProducts" column of the pandas DataFrame object named "data" for a specific subset of data. The subset is defined by the condition that the value in the "Income" column is greater than or equal to 35196 and less than 63516.
meatProduct_averageIncome = data.loc[(data['Income'] >= 35196) & (data['Income'] < 63516), 'MntMeatProducts'].sum()
```

In [140]:

```
meatProduct_averageIncome
```

Out[140]:

```
352250
```

In [141]:

```
# calculates the sum of the values in the "MntFishProducts" column of the pandas DataFrame object named "data" for a specific subset of data. The subset is defined by the condition that the value in the "Income" column is greater than or equal to 635196 and less than 113735.
MeatProduct_highIncome = data.loc[(data['Income'] >= 635196) & (data['Income'] <= 113734), 'MntMeatProducts'].sum()
```

In [142]:

```
MeatProduct_highIncome
```

Out[142]:

```
0
```

In [143]:

```
12263/364513
```

Out[143]:

```
0.0336421471936529
```

In [144]:

```
352250/364513
```

Out[144]:

```
0.9663578528063471
```

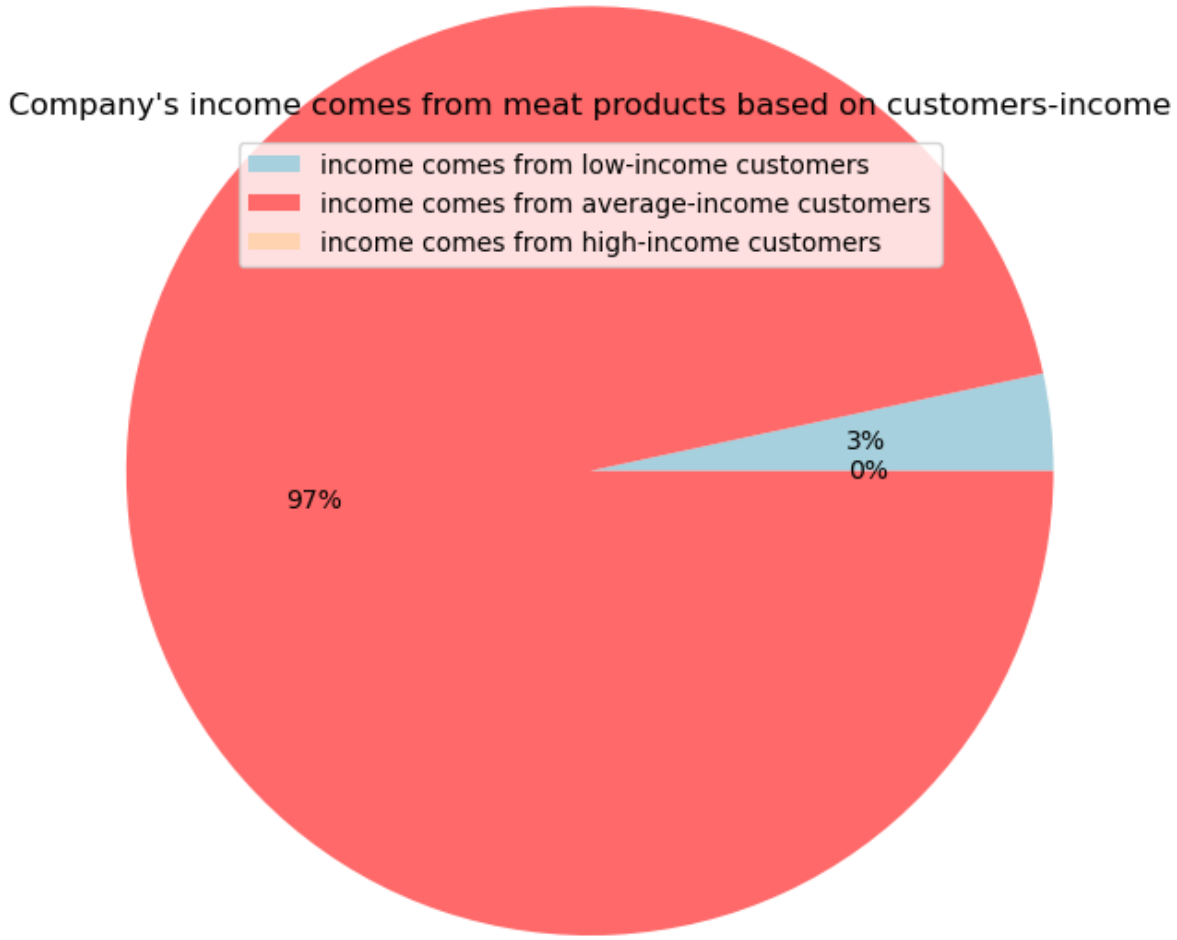
Plot for company\'s income comes from meat products based on customers-income

In [145]:

```
# Visualize the percentage of company's income comes from meat products based on customers
-income
# Define the values for the pie chart
meatProducts=[0.0336421471936529, 0.9663578528063471, 0]
# Create a pie chart with the specified values
plt.pie(meatProducts,  radius=1.7, colors=["#A6D0DD","#FF6969", "#FFD3B0"],
        rotatelabels=False, autopct= "%1.0f%%")
# Add a title to the plot
plt.title('Company\'s income comes from meat products based on customers-income')
# Add a legend to the plot with the specified labels
plt.legend(labels=["income comes from low-income customers", "income comes from average-in
come customers", "income comes from high-income customers"])
```

Out[145]:

```
<matplotlib.legend.Legend at 0x780006097fd0>
```



In [146]:

```
# calculates the sum of the values in the "MntFruits" column
data['MntFruits'].sum()
```

Out[146]:

```
58219
```

In [147]:

```
# calculates the sum of the values in the "MntMeatProducts" column of the pandas DataFrame object named "data" for a specific subset of data. The subset is defined by the condition that the value in the "Income" column is greater than or equal to 1730 and less than 35196.
fruitProduct_lowIncome = data.loc[(data['Income'] >= 1730) & (data['Income'] < 35196), 'MntFruits'].sum()
```

In [148]:

```
fruitProduct_lowIncome
```

Out[148]:

```
3264
```

In [149]:

```
# calculates the sum of the values in the "MntMeatProducts" column of the pandas DataFrame object named "data" for a specific subset of data. The subset is defined by the condition that the value in the "Income" column is greater than or equal to 35196 and less than 635196.
fruitProduct_averageIncome = data.loc[(data['Income'] >= 35196) & (data['Income'] < 635196), 'MntFruits'].sum()
```

In [150]:

```
fruitProduct_averageIncome
```

Out[150]:

```
54955
```

In [151]:

```
# calculates the sum of the values in the "MntFishProducts" column of the pandas DataFrame object named "data" for a specific subset of data. The subset is defined by the condition that the value in the "Income" column is greater than or equal to 635196 and less than 113735.
fruitProduct_highIncome = data.loc[(data['Income'] >= 635196) & (data['Income'] <= 113734), 'MntFruits'].sum()
```

In [152]:

```
fruitProduct_highIncome
```

Out[152]:

```
0
```

In [153]:

```
3264/58219
```

Out[153]:

```
0.05606417149040691
```

In [154]:

```
54955/58219
```

Out[154]:

```
0.9439358285095931
```

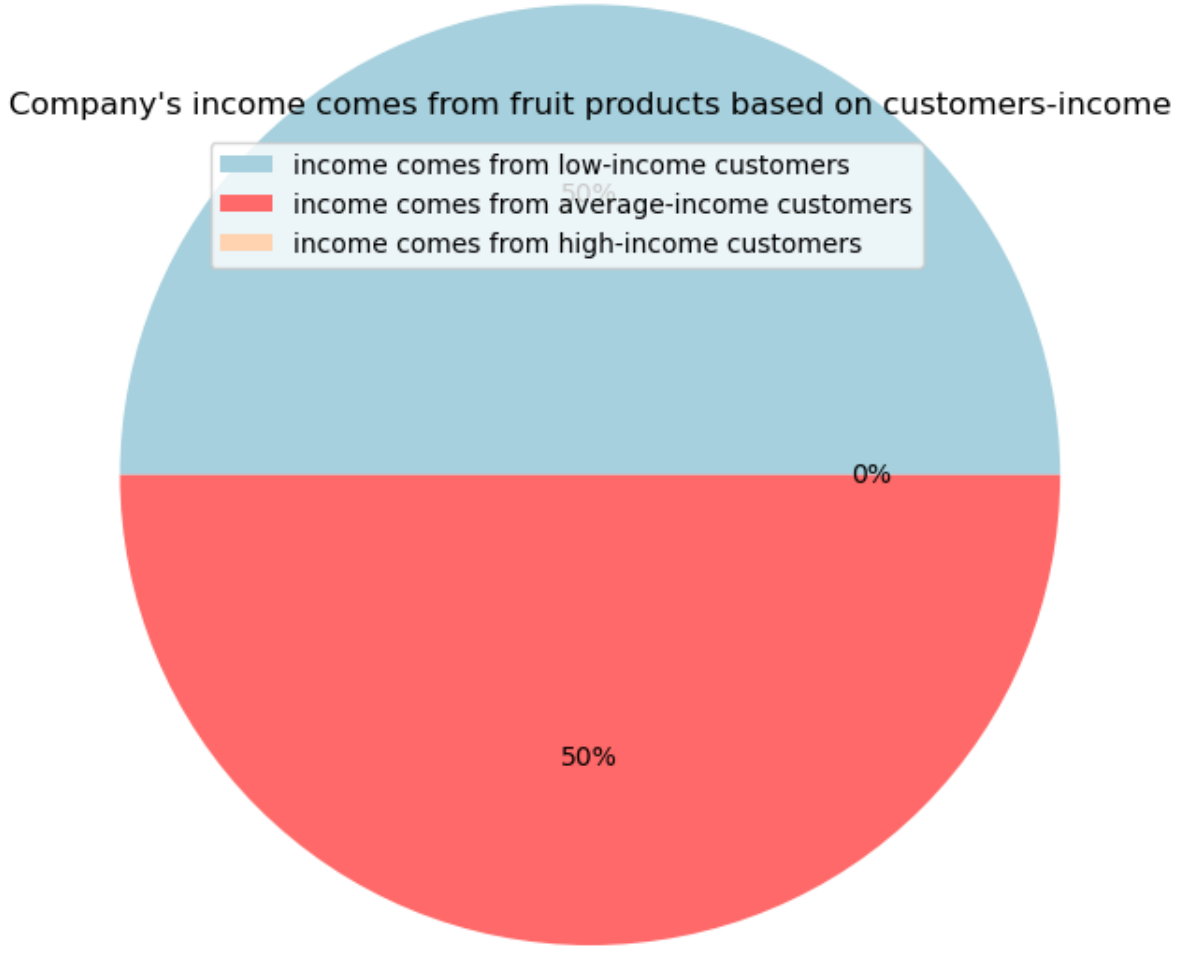
Plot for company\'s income comes from fruit products based on customers-income

In [155]:

```
# Visualize the percentage of company's income comes from fruit products based on customer
s-income
# Define the values for the pie chart
fruitProducts=[0.05606417149040691, 0.05606417149040691, 0]
# Create a pie chart with the specified values
plt.pie(fruitProducts,  radius=1.7, colors=["#A6D0DD","#FF6969", "#FFD3B0"],
        rotatelabels=False, autopct= "%1.0f%%")
# Add a title to the plot
plt.title('Company\'s income comes from fruit products based on customers-income')
# Add a legend to the plot with the specified labels
plt.legend(labels=["income comes from low-income customers", "income comes from average-in
come customers", "income comes from high-income customers"])
```

Out[155]:

```
<matplotlib.legend.Legend at 0x780006203370>
```



In [156]:

```
# calculates the sum of the values in the "MntSweetProducts" column
data['MntSweetProducts'].sum()
```

Out[156]:

```
59818
```

In [157]:

```
# calculates the sum of the values in the "MntMeatProducts" column of the pandas DataFrame object named "data" for a specific subset of data. The subset is defined by the condition that the value in the "Income" column is greater than or equal to 1730 and less than 35196.
sweatProduct_lowIncome = data.loc[(data['Income'] >= 1730) & (data['Income'] < 35196), 'MntSweetProducts'].sum()
```

In [158]:

```
sweatProduct_lowIncome
```

Out[158]:

```
3202
```

In [159]:

```
# calculates the sum of the values in the "MntMeatProducts" column of the pandas DataFrame object named "data" for a specific subset of data. The subset is defined by the condition that the value in the "Income" column is greater than or equal to 35196 and less than 635196.
sweatProduct_averageIncome = data.loc[(data['Income'] >= 35196) & (data['Income'] < 635196), 'MntSweetProducts'].sum()
```

In [160]:

```
sweatProduct_averageIncome
```

Out[160]:

```
56616
```

In [161]:

```
# calculates the sum of the values in the "MntFishProducts" column of the pandas DataFrame object named "data" for a specific subset of data. The subset is defined by the condition that the value in the "Income" column is greater than or equal to 635196 and less than 113735.
sweatProduct_highIncome = data.loc[(data['Income'] >= 635196) & (data['Income'] <= 113734), 'MntSweetProducts'].sum()
```

In [162]:

```
sweatProduct_highIncome
```

Out[162]:

```
0
```

In [163]:

```
3202/59818
```

Out[163]:

```
0.05352903808218262
```

In [164]:

```
56616/59818
```

Out[164]:

```
0.9464709619178174
```

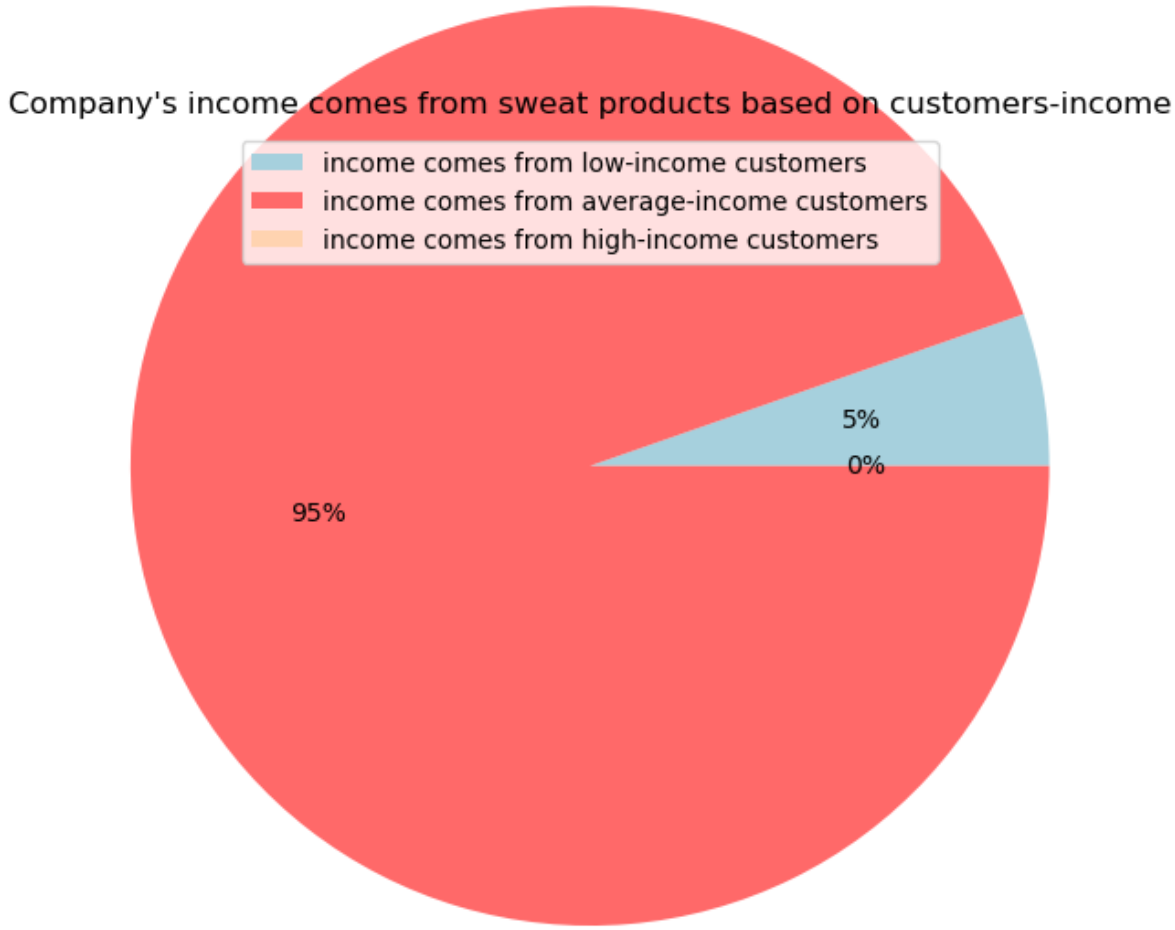
Plot for company\'s income comes from sweat products based on customers-income

In [165]:

```
# Visualize the percentage of company's income comes from sweat products based on customer
s-income
# Define the values for the pie chart
sweatProducts=[0.05352903808218262, 0.9464709619178174, 0]
# Create a pie chart with the specified values
plt.pie(sweatProducts,  radius=1.7, colors=["#A6D0DD","#FF6969", "#FFD3B0"],
        rotatelabels=False, autopct= "%1.0f%%")
# Add a title to the plot
plt.title('Company\'s income comes from sweat products based on customers-income')
# Add a legend to the plot with the specified labels
plt.legend(labels=["income comes from low-income customers", "income comes from average-in
come customers", "income comes from high-income customers"])
```

Out[165]:

```
<matplotlib.legend.Legend at 0x78000614a5f0>
```



In [166]:

```
# calculates the sum of the values in the "MntWines" column
data['MntWines'].sum()
```

Out[166]:

```
675093
```

In [167]:

```
# calculates the sum of the values in the "MntWines" column of the pandas DataFrame object named "data" for a specific subset of data. The subset is defined by the condition that th e value in the "Income" column is greater than or equal to 1730 and less than 35196.
winesProduct_lowIncome = data.loc[(data['Income'] >= 1730) & (data['Income'] < 35196), 'Mn tWines'].sum()
```

In [168]:

```
winesProduct_lowIncome
```

Out[168]:

```
11753
```

In [169]:

```
# calculates the sum of the values in the "MntWines" column of the pandas DataFrame object named "data" for a specific subset of data. The subset is defined by the condition that th e value in the "Income" column is greater than or equal to 35196 and less than 635196
winesProduct_averageIncome = data.loc[(data['Income'] >= 35196) & (data['Income'] < 63519 6), 'MntWines'].sum()
```

In [170]:

```
winesProduct_averageIncome
```

Out[170]:

```
663340
```

In [171]:

```
# calculates the sum of the values in the "MntWines" column of the pandas DataFrame object named "data" for a specific subset of data. The subset is defined by the condition that th e value in the "Income" column is greater than or equal to 635196 and less than 113735.
winesProduct_highIncome = data.loc[(data['Income'] >= 635196) & (data['Income'] <= 11373 4), 'MntWines'].sum()
```

In [172]:

```
winesProduct_highIncome
```

Out[172]:

```
0
```

In [173]:

```
11753/675093
```

Out[173]:

```
0.017409453216075416
```

In [174]:

```
663340/675093
```

Out[174]:

```
0.9825905467839245
```

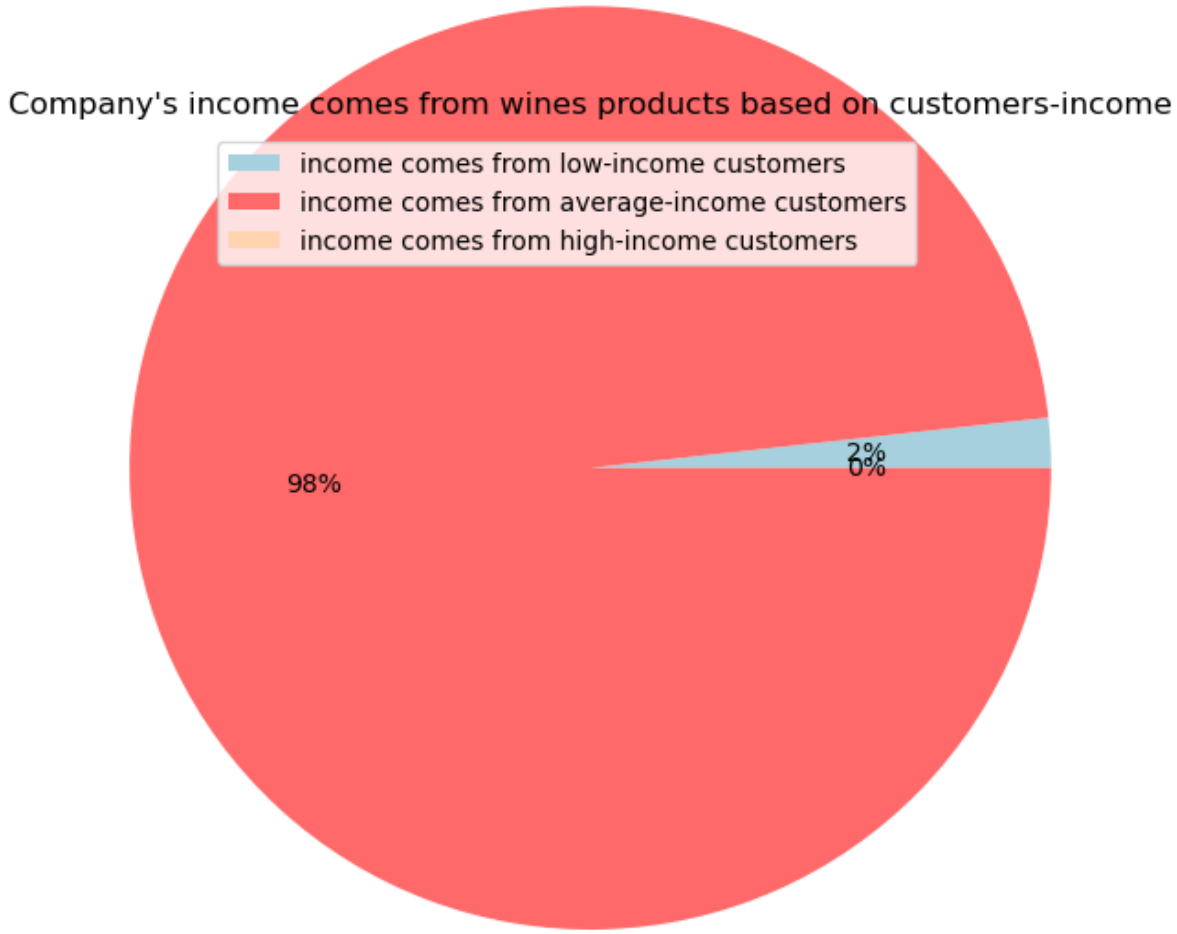
Plot for company\'s income comes from wines products based on customers-income

In [175]:

```
# Visualize the percentage of company's income comes from wines products based on customer
s-income
# Define the values for the pie chart
winesProducts=[0.017409453216075416, 0.9825905467839245, 0]
# Create a pie chart with the specified values
plt.pie(winesProducts,  radius=1.7, colors=["#A6D0DD","#FF6969", "#FFD3B0"],
        rotatelabels=False, autopct= "%1.0f%%")
# Add a title to the plot
plt.title('Company\'s income comes from wines products based on customers-income')
# Add a legend to the plot with the specified labels
plt.legend(labels=["income comes from low-income customers", "income comes from average-in
come customers", "income comes from high-income customers"])
```

Out[175]:

```
<matplotlib.legend.Legend at 0x780005fabd30>
```



In [176]:

```
# calculates the sum of the values in the "MntGoldProds" column
data['MntGoldProds'].sum()
```

Out[176]:

```
97146
```

In [177]:

```
# calculates the sum of the values in the "MntGoldProds" column of the pandas DataFrame ob
ject named "data" for a specific subset of data. The subset is defined by the condition th
at the value in the "Income" column is greater than or equal to 1730 and less than 35196.
goldProduct_lowIncome = data.loc[(data['Income'] >= 1730) & (data['Income'] < 35196), 'Mnt
GoldProds'].sum()
```

In [178]:

```
goldProduct_lowIncome
```

Out[178]:

8951

In [179]:

```
# calculates the sum of the values in the "MntGoldProds" column of the pandas DataFrame ob
ject named "data" for a specific subset of data. The subset is defined by the condition th
at the value in the "Income" column is greater than or equal to 35196 and less than 635196
goldProduct_averageIncome = data.loc[(data['Income'] >= 35196) & (data['Income'] < 63519
6), 'MntGoldProds'].sum()
```

In [180]:

```
goldProduct_averageIncome
```

Out[180]:

88195

In [181]:

```
# calculates the sum of the values in the "MntGoldProds" column of the pandas DataFrame ob
ject named "data" for a specific subset of data. The subset is defined by the condition th
at the value in the "Income" column is greater than or equal to 635196 and less than 11373
5.
goldProduct_highIncome = data.loc[(data['Income'] >= 635196) & (data['Income'] <= 113734),
'MntGoldProds'].sum()
```

In [182]:

```
goldProduct_highIncome
```

Out[182]:

0

In [183]:

```
8951/97146
```

Out[183]:

0.09213966606962716

In [184]:

```
88195/97146
```

Out[184]:

```
0.9078603339303728
```

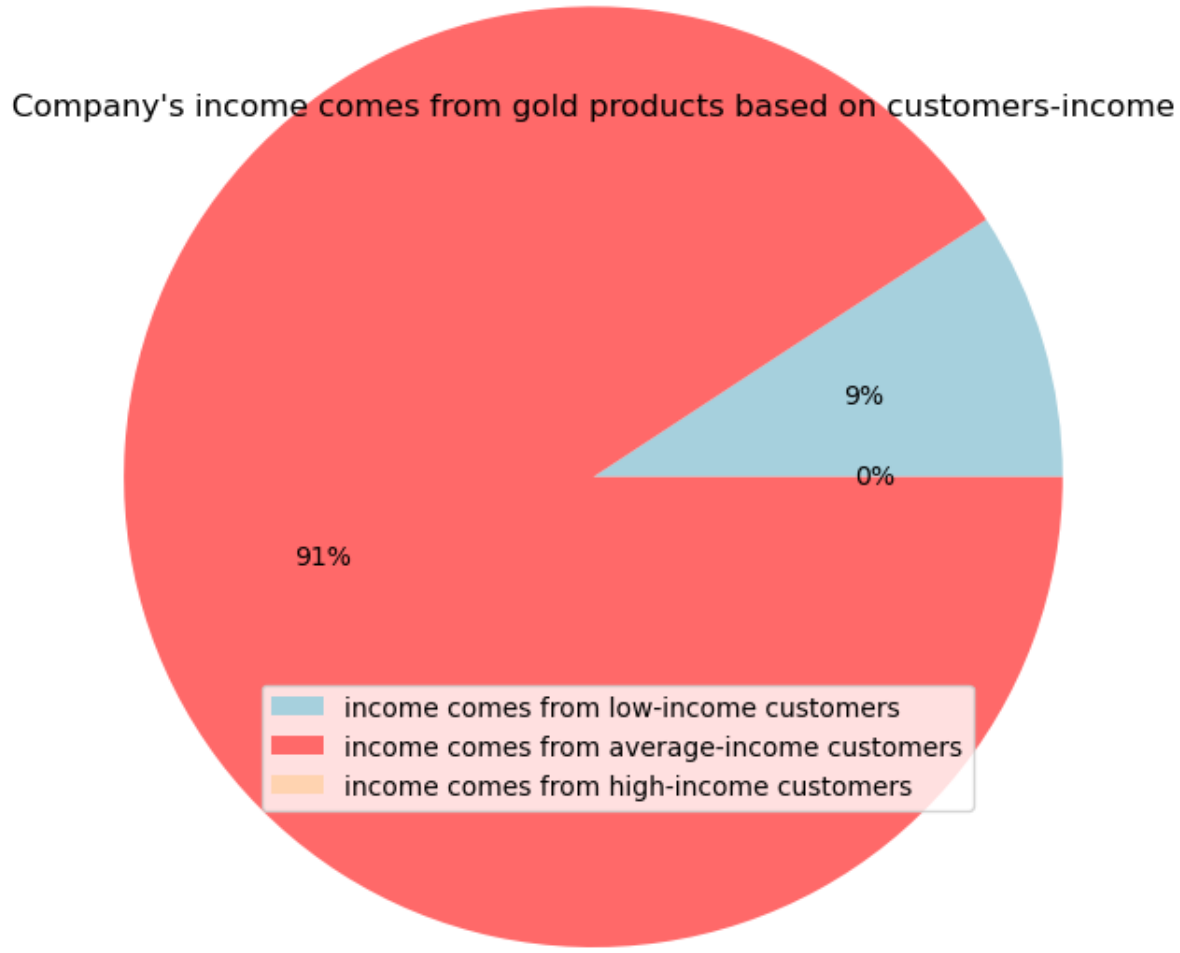
Plot for company\'s income comes from gold products based on customers-income

In [185]:

```
# Visualize the percentage of company's income comes from gold products based on customers
-income
# Define the values for the pie chart
goldProducts=[0.09213966606962716, 0.9078603339303728, 0]
# Create a pie chart with the specified values
plt.pie(goldProducts,  radius=1.7, colors=["#A6D0DD","#FF6969", "#FFD3B0"],
        rotatelabels=False, autopct= "%1.0f%%")
# Add a title to the plot
plt.title('Company\'s income comes from gold products based on customers-income')
# Add a legend to the plot with the specified labels
plt.legend(labels=["income comes from low-income customers", "income comes from average-in
come customers", "income comes from high-income customers"])
```

Out[185]:

```
<matplotlib.legend.Legend at 0x78000600ba60>
```



No Products from customers with high-income

In [186]:

```
# calculates the sum of the values in the "NumStorePurchases" column
data["NumStorePurchases"].sum()
```

Out[186]:

12841

In [187]:

```
# calculates the sum of the values in the "NumStorePurchases" column of the pandas DataFra
me object named "data" for a specific subset of data. The subset is defined by the conditi
on that the value in the "Income" column is greater than or equal to 1730 and less than 35
196.
StorePurchases_lowIncome = data.loc[(data['Income'] >= 1730) & (data['Income'] < 35196),
'NumStorePurchases'].sum()
```

In [188]:

```
StorePurchases_lowIncome
```

Out[188]:

1627

In [189]:

```
# calculates the sum of the values in the "NumStorePurchases" column of the pandas DataFra
me object named "data" for a specific subset of data. The subset is defined by the conditi
on that the value in the "Income" column is greater than or equal to 35196 and less than 6
35196
StorePurchases_averageIncome = data.loc[(data['Income'] >= 35196) & (data['Income'] < 6351
96), 'NumStorePurchases'].sum()
```

In [190]:

```
StorePurchases_averageIncome
```

Out[190]:

11214

In [191]:

```
# calculates the sum of the values in the "NumStorePurchases" column of the pandas DataFra
me object named "data" for a specific subset of data. The subset is defined by the conditi
on that the value in the "Income" column is greater than or equal to 635196 and less than
113735.
StorePurchases_highIncome = data.loc[(data['Income'] >= 635196) & (data['Income'] <= 11373
4), 'NumStorePurchases'].sum()
```

In [192]:

```
StorePurchases_highIncome
```

Out[192]:

```
0
```

In [193]:

```
1627/12841
```

Out[193]:

```
0.1267035277626353
```

In [194]:

```
11214/12841
```

Out[194]:

```
0.8732964722373647
```

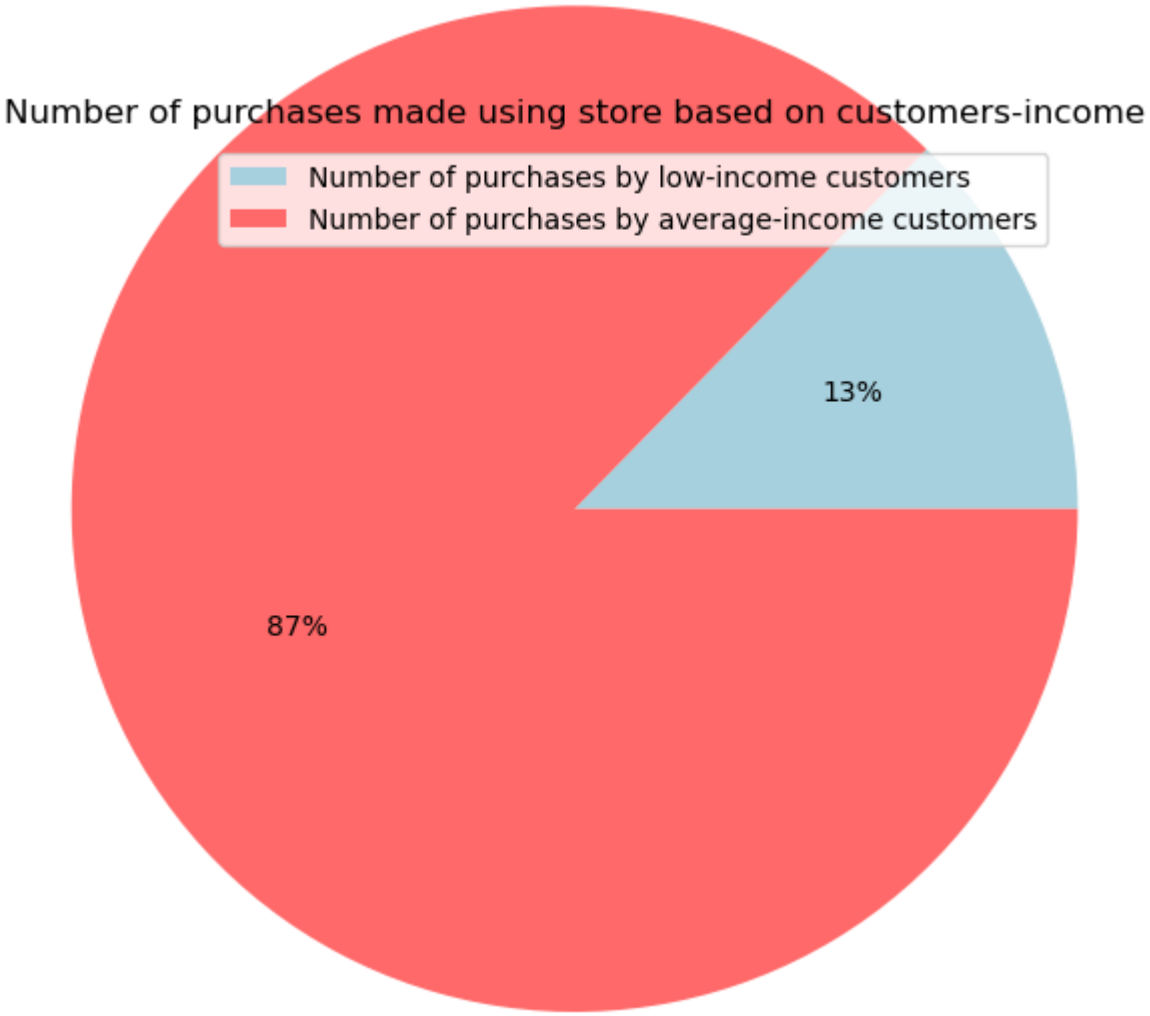
Plot for number of purchases made using store based on customers-income

In [195]:

```
# Visualize the percentage of number of purchases made using store based on customers-inco
me
# Define the values for the pie chart
StorePurchases=[0.1267035277626353, 0.8732964722373647]
# Create a pie chart with the specified values
plt.pie(StorePurchases,  radius=1.7, colors=["#A6D0DD","#FF6969"],
        rotatelabels=False, autopct= "%1.0f%%")
# Add a title to the plot
plt.title('Number of purchases made using store based on customers-income')
# Add a legend to the plot with the specified labels
plt.legend(labels=["Number of purchases by low-income customers", "Number of purchases by
average-income customers"])
```

Out[195]:

```
<matplotlib.legend.Legend at 0x780005e76440>
```



In [196]:

```
# calculates the sum of the values in the "NumCatalogPurchases" column
data["NumCatalogPurchases"].sum()
```

Out[196]:

5833

In [197]:

```
# calculates the sum of the values in the "NumCatalogPurchases" column of the pandas DataF
rame object named "data" for a specific subset of data. The subset is defined by the condi
tion that the value in the "Income" column is greater than or equal to 1730 and less than
35196.
catalogPurchases_lowIncome = data.loc[(data['Income'] >= 1730) & (data['Income'] < 35196),
'NumCatalogPurchases'].sum()
```

In [198]:

```
catalogPurchases_lowIncome
```

Out[198]:

261

In [199]:

```
# calculates the sum of the values in the "NumCatalogPurchases" column of the pandas DataF
rame object named "data" for a specific subset of data. The subset is defined by the condi
tion that the value in the "Income" column is greater than or equal to 35196 and less than
635196
catalogPurchases_averageIncome = data.loc[(data['Income'] >= 35196) & (data['Income'] < 63
5196), 'NumCatalogPurchases'].sum()
```

In [200]:

```
catalogPurchases_averageIncome
```

Out[200]:

5572

In [201]:

```
261/5833
```

Out[201]:

0.044745414023658496

In [202]:

```
5572/5833
```

Out[202]:

0.9552545859763415

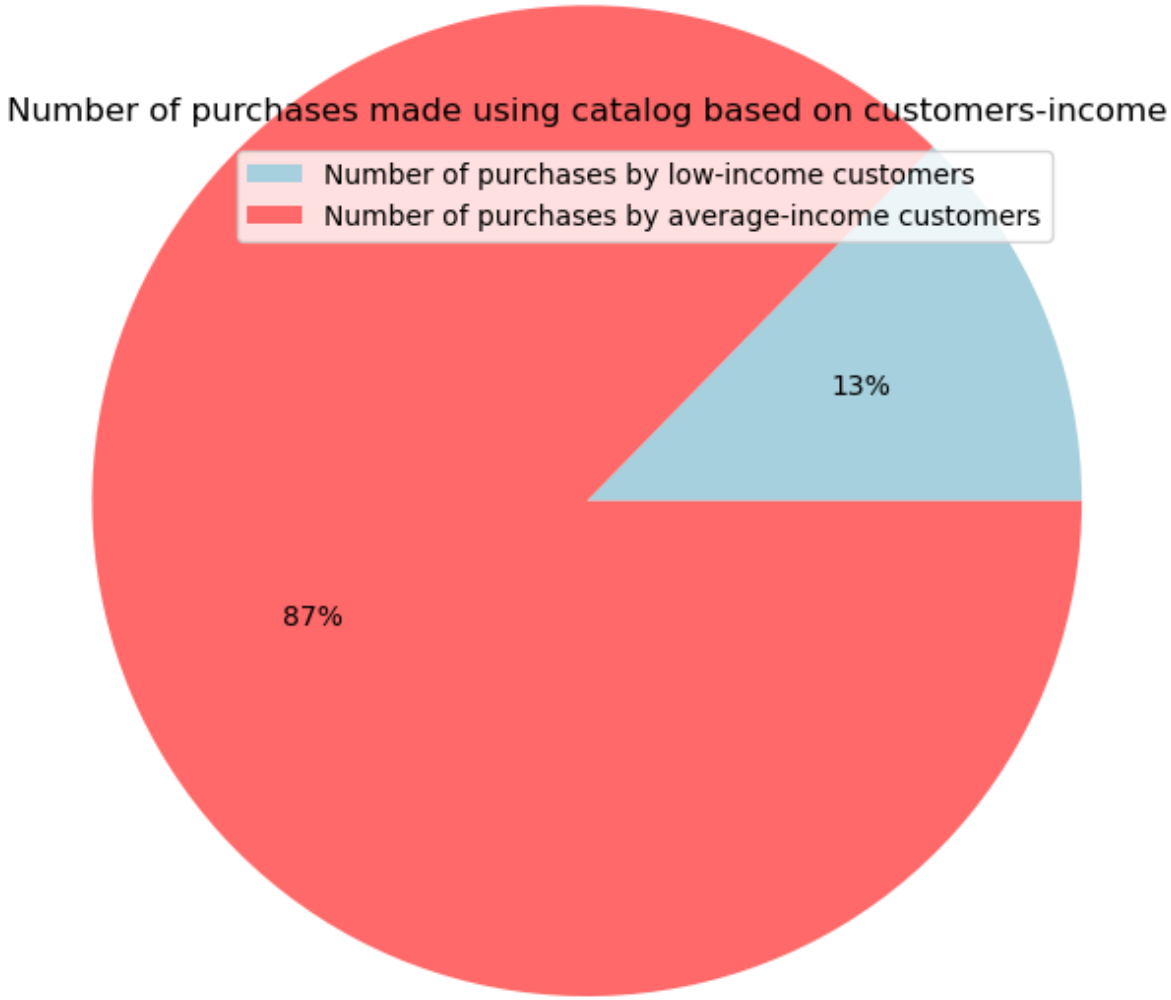
Plot for number of purchases made using catalog based on customers-income

In [203]:

```
# Visualize the percentage of number of purchases made using catalog based on customers-in
come
# Define the values for the pie chart
CatalogPurchases=[0.044745414023658496, 0.9552545859763415]
# Create a pie chart with the specified values
plt.pie(StorePurchases,  radius=1.7, colors=["#A6D0DD","#FF6969"],
        rotatelabels=False, autopct= "%1.0f%%")
# Add a title to the plot
plt.title('Number of purchases made using catalog based on customers-income')
# Add a legend to the plot with the specified labels
plt.legend(labels=["Number of purchases by low-income customers", "Number of purchases by
average-income customers"])
```

Out[203]:

```
<matplotlib.legend.Legend at 0x780005ee39a0>
```



In [204]:

```
# calculates the sum of the values in the "NumWebPurchases" column
data["NumWebPurchases"].sum()
```

Out[204]:

```
9042
```

In [205]:

```
# calculates the sum of the values in the "NumWebPurchases" column of the pandas DataFrame object named "data" for a specific subset of data. The subset is defined by the condition that the value in the "Income" column is greater than or equal to 1730 and less than 35196.
webPurchases_lowIncome = data.loc[(data['Income'] >= 1730) & (data['Income'] < 35196), 'NumWebPurchases'].sum()
```

In [206]:

```
webPurchases_lowIncome
```

Out[206]:

```
1100
```

In [207]:

```
# calculates the sum of the values in the "NumWebPurchases" column of the pandas DataFrame object named "data" for a specific subset of data. The subset is defined by the condition that the value in the "Income" column is greater than or equal to 35196 and less than 635196
webPurchases_averageIncome = data.loc[(data['Income'] >= 35196) & (data['Income'] < 635196), 'NumWebPurchases'].sum()
```

In [208]:

```
webPurchases_averageIncome
```

Out[208]:

```
7942
```

In [209]:

```
1100/9042
```

Out[209]:

```
0.12165450121654502
```

In [210]:

```
7942/9042
```

Out[210]:

```
0.878345498783455
```

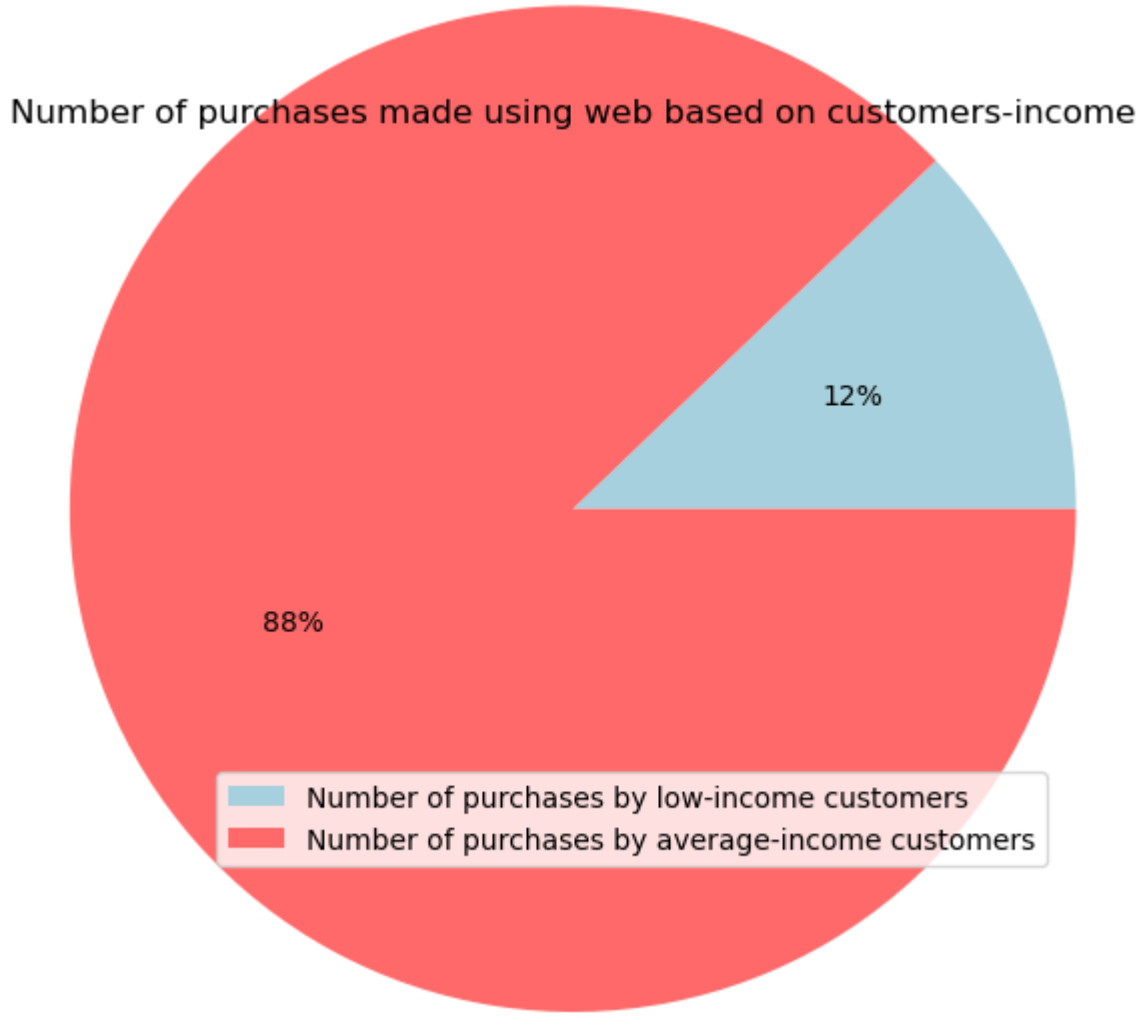
Plot for number of purchases made using web based on customers-income

In [211]:

```
# Visualize the percentage of number of purchases made using web based on customers-income
# Define the values for the pie chart
WebPurchases=[0.12165450121654502, 0.878345498783455]
# Create a pie chart with the specified values
plt.pie(WebPurchases,  radius=1.7, colors=["#A6D0DD","#FF6969"],
        rotatelabels=False, autopct= "%1.0f%%")
# Add a title to the plot
plt.title('Number of purchases made using web based on customers-income')
# Add a legend to the plot with the specified labels
plt.legend(labels=["Number of purchases by low-income customers", "Number of purchases by
average-income customers"])
```

Out[211]:

```
<matplotlib.legend.Legend at 0x780005d7fb80>
```



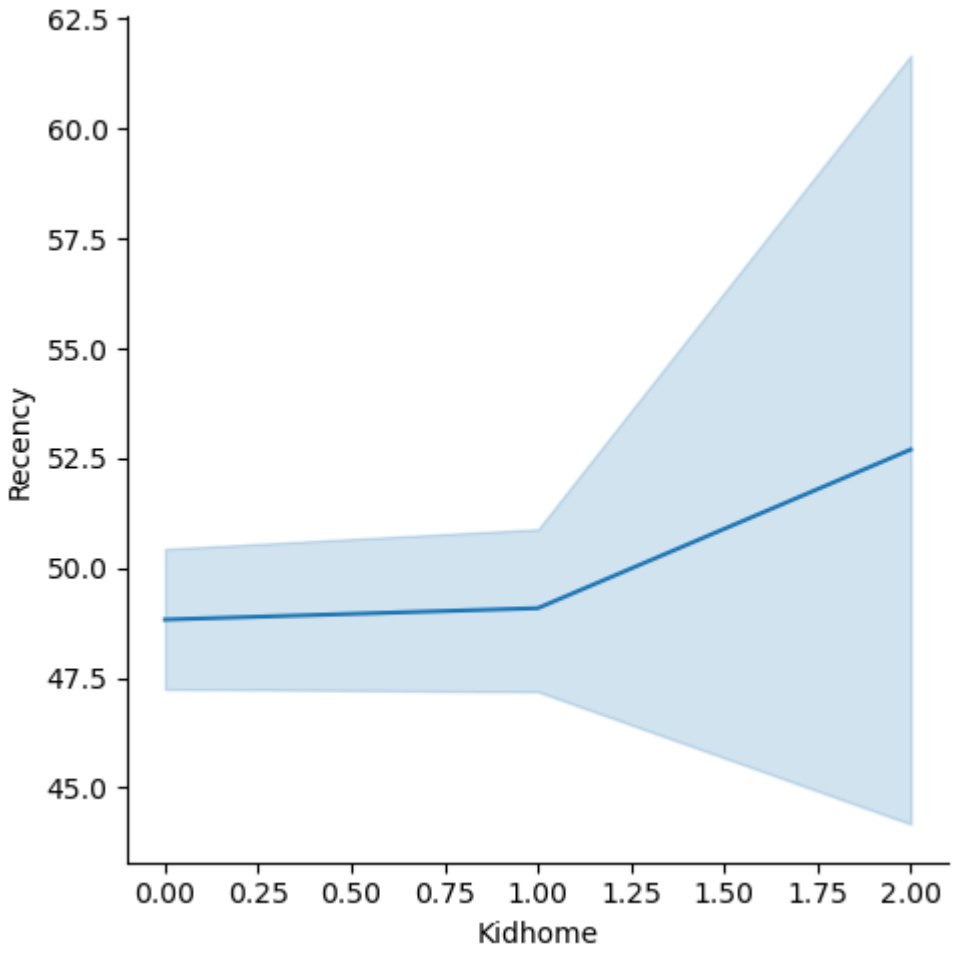
The relationship between number of kids in home and the number of days since the last purchases

In [212]:

```
# Create a line plot using seaborn to visualize the relationship between "Kidhome" and "Recency"
sns.relplot(data=data, x="Kidhome", y="Recency", kind="line")
```

Out[212]:

```
<seaborn.axisgrid.FacetGrid at 0x780005eb0520>
```



Insight :When there was no kids in the home, the number of days since the last purchase was approximately equal to the number of days when the number of kids was 1

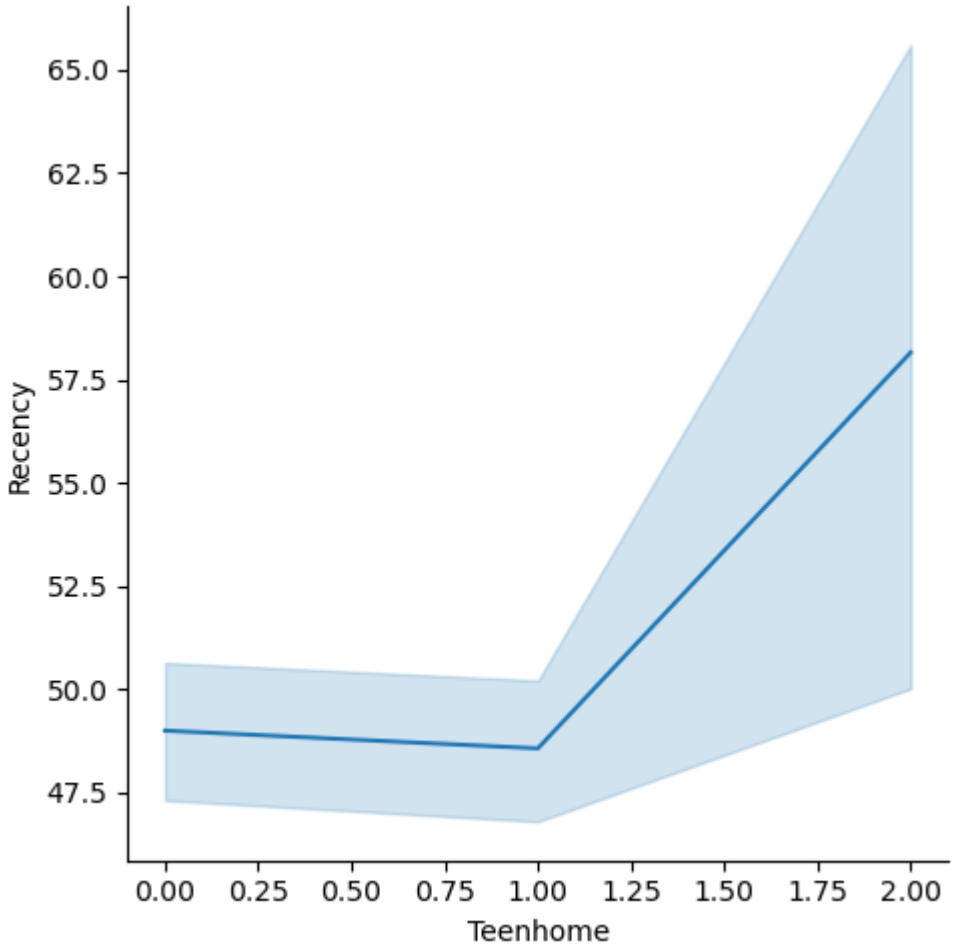
When the number of kids in the home increased to 2, the number of days since the last purchase increased

In [213]:

```
# Create a line plot using seaborn to visualize the relationship between "Teenhome" and "Recency"
sns.relplot(data=data, x="Teenhome", y="Recency", kind="line")
```

Out[213]:

```
<seaborn.axisgrid.FacetGrid at 0x780005f46dd0>
```



Insight :When the number of teens in home was 1, the number of days since the last purchase was less than the number of days since the last purchase when there were no teens in the home

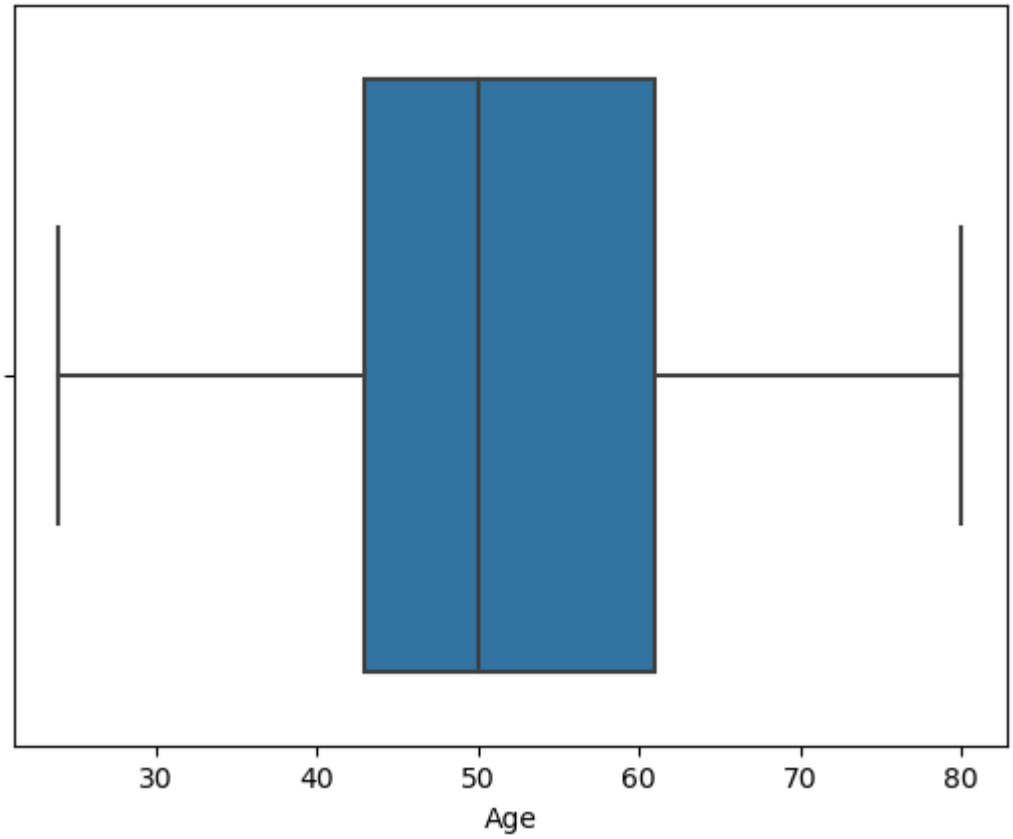
When the number of teens in the home increased to 2, the number of days since the last purchase increased

In [214]:

```
# Create a box plot using seaborn to visualize the distribution of Age in dataframe
sns.boxplot(x=data["Age"])
```

Out[214]:

```
<Axes: xlabel='Age'>
```



In [215]:

```
# Calculate the third and first quartiles of a single variable
q3, q1 = np.percentile(data['Age'], [75, 25])
```

In [216]:

```
# 25% from data
q1
```

Out[216]:

```
43.0
```

In [217]:

```
# 75% from data
q3
```

Out[217]:

```
61.0
```

In [218]:

```
# the maximum value in Age column
data["Age"].max()
```

Out[218]:

```
80
```

In [219]:

```
# the minimum value in Age column
data["Age"].min()
```

Out[219]:

```
24
```

first range of ages from 24 to 43

second range of ages from 44 to 61

third range of ages from 62 to 80

In [220]:

```
# Convert a pandas DataFrame column to a numpy array
Age_array = data['Age'].values
```

In [221]:

```
# counts the number of values in a numpy array "Age_array" that fall within a specified ra
nge (between 24 and 44, inclusive).
counter_low_age = 0
for i_low_age in Age_array:
    if (i_low_age >=24) & (i_low_age < 44):
        counter_low_age +=1
    else:
        continue
```

In [222]:

```
# number of customers whose ages between 24 and 43
counter_low_age
```

Out[222]:

```
597
```

In [223]:

```
# counts the number of values in a numpy array "Age_array" that fall within a specified ra
nge (between 44 and 62, inclusive).
counter_average_age = 0
for i_average_age in Age_array:
    if (i_average_age >=44) & (i_average_age < 62):
        counter_average_age +=1
    else:
        continue
```

In [224]:

```
# number of customers whose ages between 44 and 61
counter_average_age
```

Out[224]:

```
1100
```

In [225]:

```
# counts the number of values in a numpy array "Age_array" that fall within a specified ra
nge (between 62 and 81, inclusive).
counter_high_age = 0
for i_high_age in Age_array:
    if (i_high_age >=62) & (i_high_age < 81):
        counter_high_age +=1
    else:
        continue
```

In [226]:

```
# number of customers whose ages between 62 and 80
counter_high_age
```

Out[226]:

```
508
```

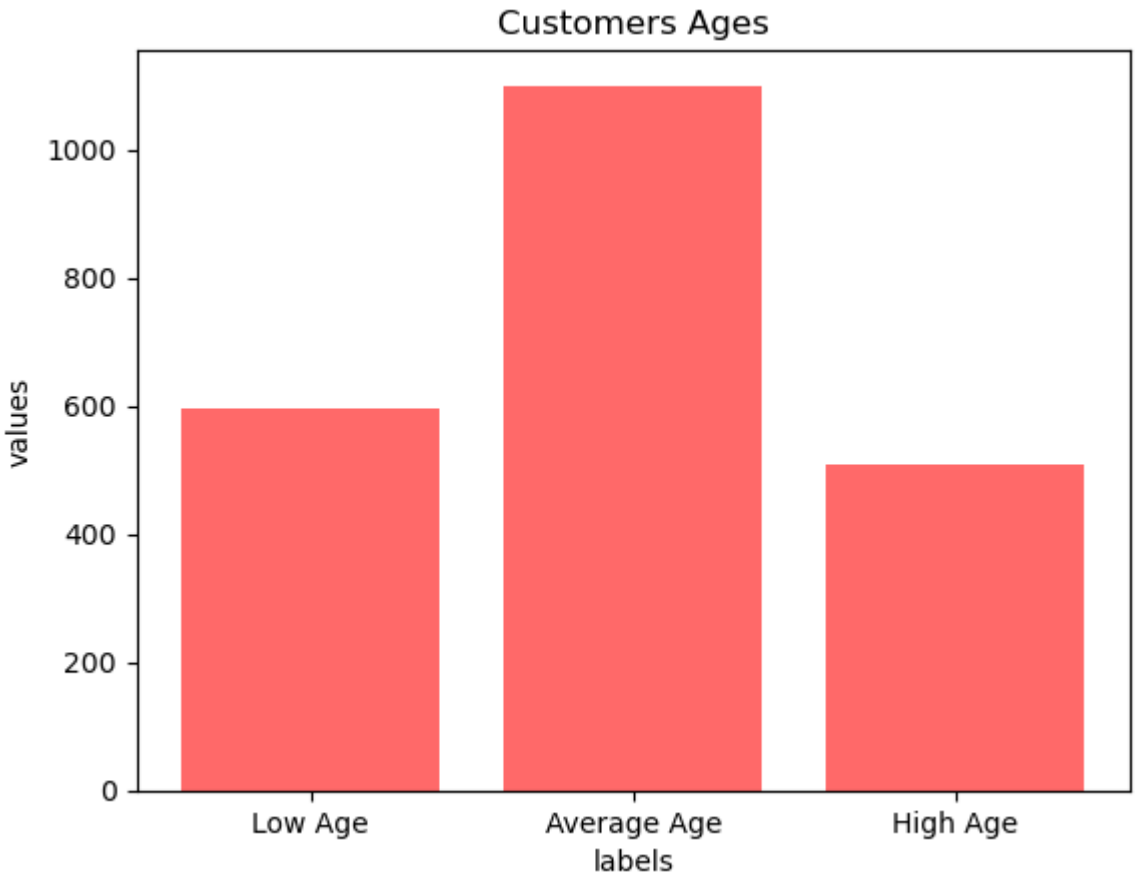
Plot the customers ages

In [227]:

```
# Define the values for the bar chart
customers_Age = [597, 1100, 508]
# Define the labels for the x-axis categories
labels = ["Low Age", "Average Age", "High Age"]
# Set the x-tick labels to the specified categories
plt.xticks(range(len(customers_Age)), labels)
# Add a label to the x-axis
plt.xlabel('labels')
# Add a label to the y-axis
plt.ylabel('values')
# Add a title to the plot
plt.title('Customers Ages')
# Create a bar chart with the specified values
plt.bar(range(len(customers_Age)), customers_Age, color="#FF6969")
```

Out[227]:

```
<BarContainer object of 3 artists>
```



In [228]:

```
# Calculate the sum of a numeric column in a pandas DataFrame
data['NumWebVisitsMonth'].sum()
```

Out[228]:

```
11768
```

In [229]:

```
# The number of visits to the site in the last month for customers whose ages range from 24 to 43 years
web_visits_lowAge = data.loc[(data['Age'] >= 24) & (data['Age'] < 44), 'NumWebVisitsMonth'].sum()
```

In [230]:

```
web_visits_lowAge
```

Out[230]:

3290

In [231]:

```
# The number of visits to the site in the last month for customers whose ages range from 44 to 61 years
web_visits_averageAge = data.loc[(data['Age'] >= 44) & (data['Age'] < 62), 'NumWebVisitsMonth'].sum()
```

In [232]:

```
web_visits_averageAge
```

Out[232]:

6050

In [233]:

```
# The number of visits to the site in the last month for customers whose ages range from 62 to 80 years
web_visits_highAge = data.loc[(data['Age'] >= 62) & (data['Age'] < 81), 'NumWebVisitsMonth'].sum()
```

In [234]:

```
web_visits_highAge
```

Out[234]:

2428

In [235]:

```
3290/11768
```

Out[235]:

0.27957171991842283

In [236]:

```
6050/11768
```

Out[236]:

```
0.5141060503059144
```

In [237]:

```
2428/11768
```

Out[237]:

```
0.2063222297756628
```

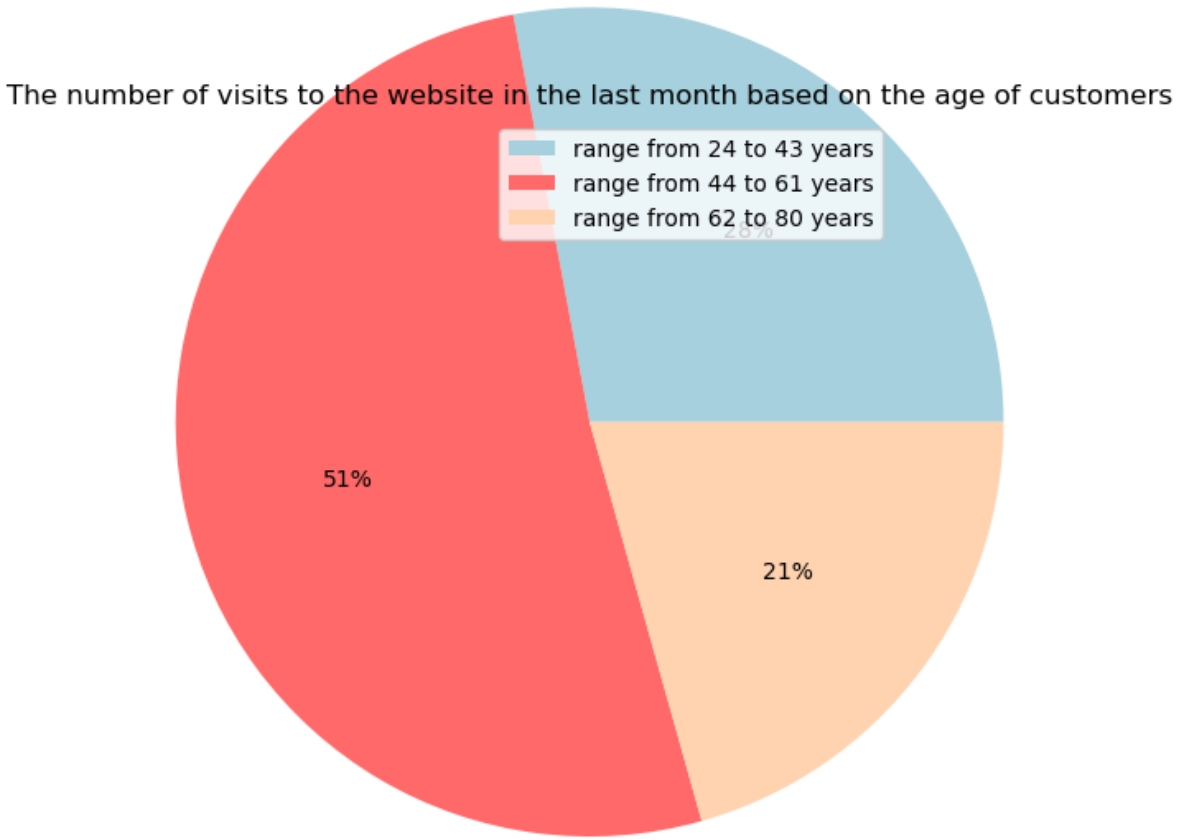
The number of visits to the website in the last month based on the age of customers

In [238]:

```
# Visualize the percentage of website visits in the last month based on the age of customers using a pie chart
# Define the values for the pie chart
site_visits_ages=[0.27957171991842283, 0.5141060503059144, 0.2063222297756628]
# Create a pie chart with the specified values
plt.pie(site_visits_ages,  radius=1.7, colors=["#A6D0DD","#FF6969", "#FFD3B0"],
        rotatelabels=False, autopct= "%1.0f%%")
# Add a title to the plot
plt.title('The number of visits to the website in the last month based on the age of customers')
# Add a legend to the plot with the specified labels
plt.legend(labels=["range from 24 to 43 years", "range from 44 to 61 years", "range from 62 to 80 years"])
```

Out[238]:

```
<matplotlib.legend.Legend at 0x780005bb7340>
```



Most of the customers who visited the site in the last month are those whose ages range from 44 to 61 years

The number of purchases made through the catalog based on the ages of customers

In [239]:

```
# Calculate the sum of a "NumCatalogPurchases" column in a DataFrame
data['NumCatalogPurchases'].sum()
```

Out[239]:

```
5833
```

In [240]:

```
# The number of purchases made with catalog from customers whose ages range from 24 to 43 years
catalog_purchases_lowAges = data.loc[(data['Age'] >= 24) & (data['Age'] < 44), 'NumCatalog Purchases'].sum()
```

In [241]:

```
catalog_purchases_lowAges
```

Out[241]:

```
1365
```

In [242]:

```
# The number of purchases made with catalog from customers whose ages range from 44 to 61 years
catalog_purchases_averageAges = data.loc[(data['Age'] >= 44) & (data['Age'] < 62), 'NumCat alogPurchases'].sum()
```

In [243]:

```
catalog_purchases_averageAges
```

Out[243]:

```
2765
```

In [244]:

```
# The number of purchases made with catalog from customers whose ages range from 62 to 80 years
catalog_purchases_highAges = data.loc[(data['Age'] >= 62) & (data['Age'] < 81), 'NumCatalo gPurchases'].sum()
```

In [245]:

```
catalog_purchases_highAges
```

Out[245]:

```
1703
```

In [246]:

```
1365/5833
```

Out[246]:

```
0.23401337219269672
```

In [247]:

```
2765/5833
```

Out[247]:

```
0.4740270872621293
```

In [248]:

```
1703/5833
```

Out[248]:

```
0.29195954054517403
```

The number of purchases made through the catalog based on the ages of customers

23% of customers who make purchases through the catalog are between the ages of 24 and 43 years

47% of customers who make purchases through the catalog are between the ages of 44 and 61 years

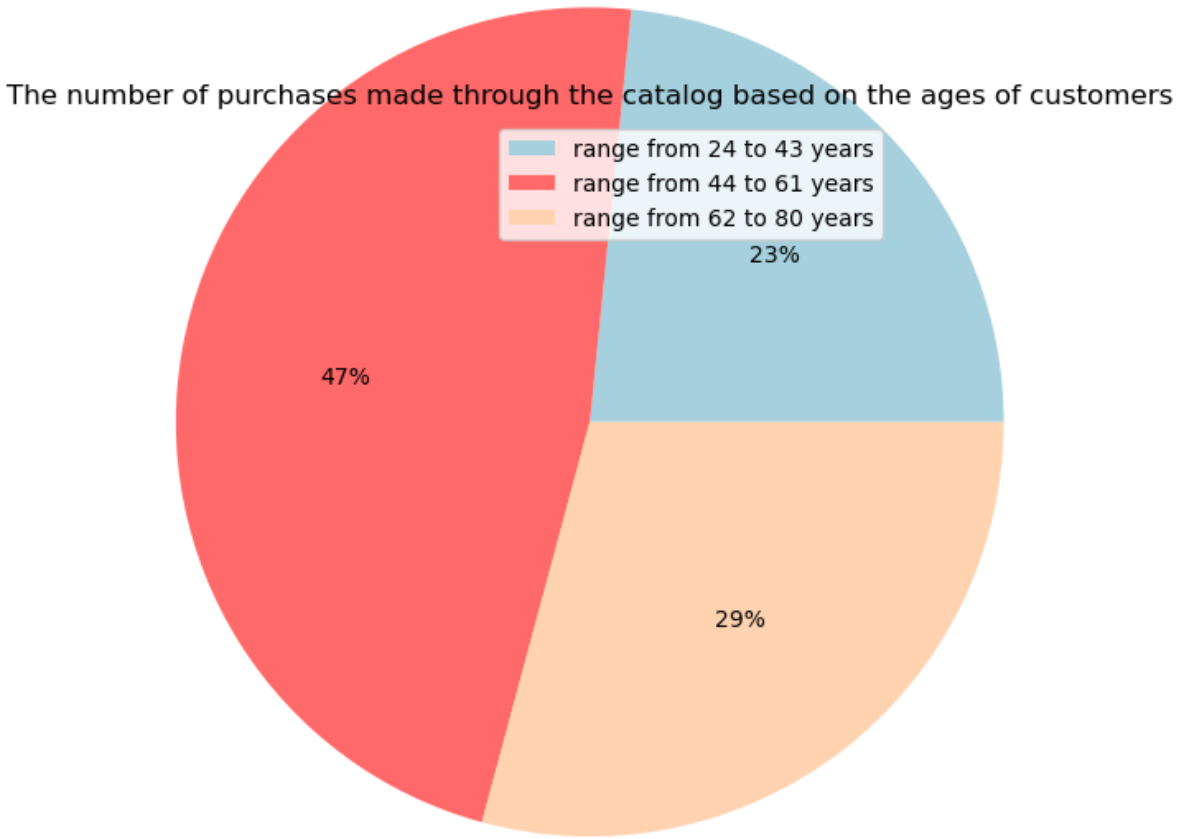
29% of customers who make purchases through the catalog are between the ages of 62 and 80 years

In [249]:

```
# Visualize the percentage of catalog purchases based on the age of customers using a pie
chart
# Define the values for the pie chart
catalog_purchases_ages = [0.23401337219269672, 0.4740270872621293, 0.29195954054517403]
# Create a pie chart with the specified values
plt.pie(catalog_purchases_ages,  radius=1.7, colors=["#A6D0DD","#FF6969", "#FFD3B0"],
        rotatelabels=False, autopct= "%1.0f%%")
# Add a title to the plot
plt.title('The number of purchases made through the catalog based on the ages of customer
s')
# Add a legend to the plot with the specified label
plt.legend(labels=["range from 24 to 43 years", "range from 44 to 61 years", "range from 6
2 to 80 years"])
```

Out[249]:

```
<matplotlib.legend.Legend at 0x780005be8190>
```



In [250]:

```
# Calculate the sum of a "NumStorePurchases" column in a DataFrame
data['NumStorePurchases'].sum()
```

Out[250]:

```
12841
```

In [251]:

```
# The number of purchases made with store from customers whose ages range from 24 to 43 years
store_purchases_lowAges = data.loc[(data['Age'] >= 24) & (data['Age'] < 44), 'NumStorePurchases'].sum()
```

In [252]:

```
store_purchases_lowAges
```

Out[252]:

3187

In [253]:

```
# The number of purchases made with store from customers whose ages range from 44 to 61 years
store_purchases_averageAges = data.loc[(data['Age'] >= 44) & (data['Age'] < 62), 'NumStorePurchases'].sum()
```

In [254]:

```
store_purchases_averageAges
```

Out[254]:

6360

In [255]:

```
# The number of purchases made with store from customers whose ages range from 62 to 80 years
store_purchases_highAges = data.loc[(data['Age'] >= 62) & (data['Age'] < 81), 'NumStorePurchases'].sum()
```

In [256]:

```
store_purchases_highAges
```

Out[256]:

3294

In [257]:

```
3187/12841
```

Out[257]:

0.2481893933494276

In [258]:

```
6360/12841
```

Out[258]:

```
0.49528852893076863
```

In [259]:

```
3294/12841
```

Out[259]:

```
0.25652207771980373
```

The number of purchases made through the store based on the ages of customer

24% of customers who make purchases through the store are between the ages of 24 and 43 years

50% of customers who make purchases through the store are between the ages of 44 and 61 years

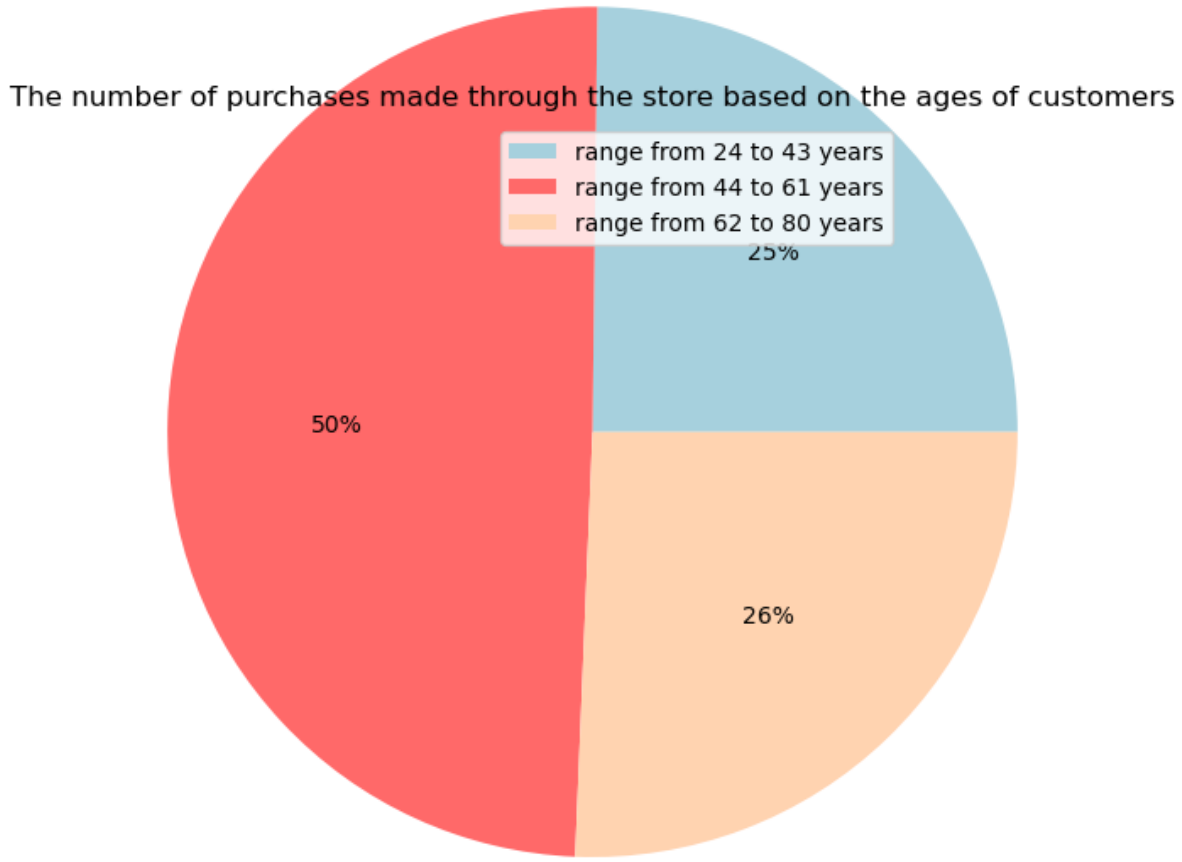
26% of customers who make purchases through the store are between the ages of 62 and 80 years

In [260]:

```
# Visualize the percentage of store purchases based on the age of customers using a pie ch
art
# Define the values for the pie chart
store_purchases_ages = [0.2481893933494276, 0.49528852893076863, 0.25652207771980373]
# Create a pie chart with the specified values
plt.pie(store_purchases_ages,  radius=1.7, colors=["#A6D0DD","#FF6969", "#FFD3B0"],
        rotatelabels=False, autopct= "%1.0f%%")
# Add a title to the plot
plt.title('The number of purchases made through the store based on the ages of customers')
# Add a legend to the plot with the specified labels
plt.legend(labels=["range from 24 to 43 years", "range from 44 to 61 years", "range from 6
2 to 80 years"])
```

Out[260]:

```
<matplotlib.legend.Legend at 0x780005ac7af0>
```



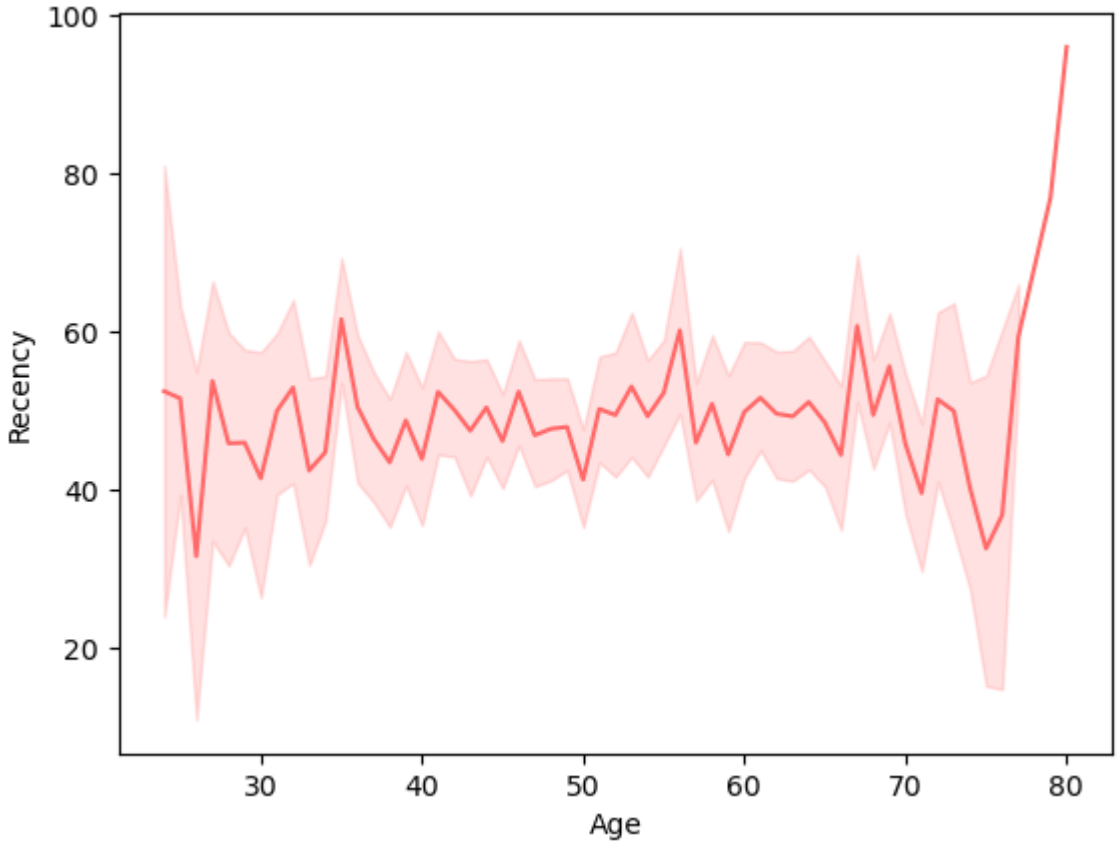
The relationship between the age of customers and the number of days since the last purchase (Recency)

In [261]:

```
# Create a line plot using seaborn to visualize the relationship between "Age" and "Recenc
y"
sns.lineplot(x='Age', y='Recency',color='#FF6969', data=data)
```

Out[261]:

```
<Axes: xlabel='Age', ylabel='Recency'>
```



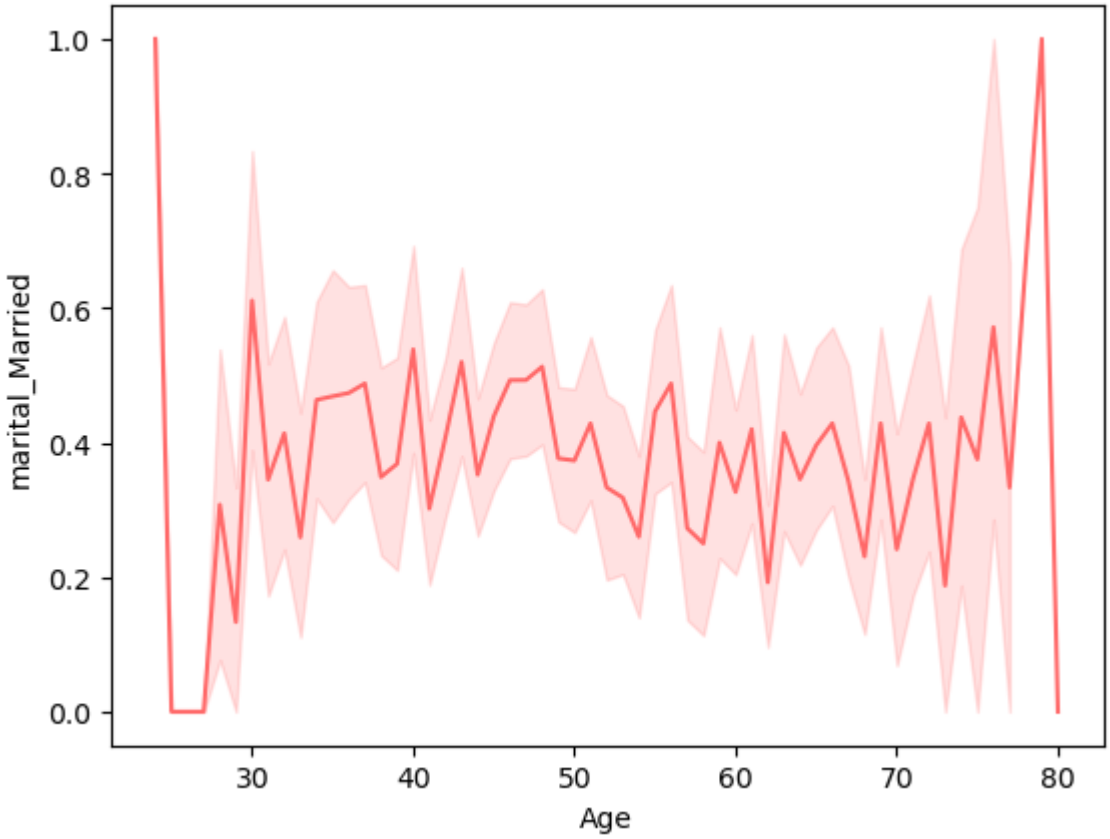
The relationship between the Married customers and the Married-status

In [262]:

```
# Create a line plot using seaborn to visualize the relationship between "Age" and "marital_Married"
sns.lineplot(x='Age', y='marital_Married',color='#FF6969', data=data)
```

Out[262]:

```
<Axes: xlabel='Age', ylabel='marital_Married'>
```



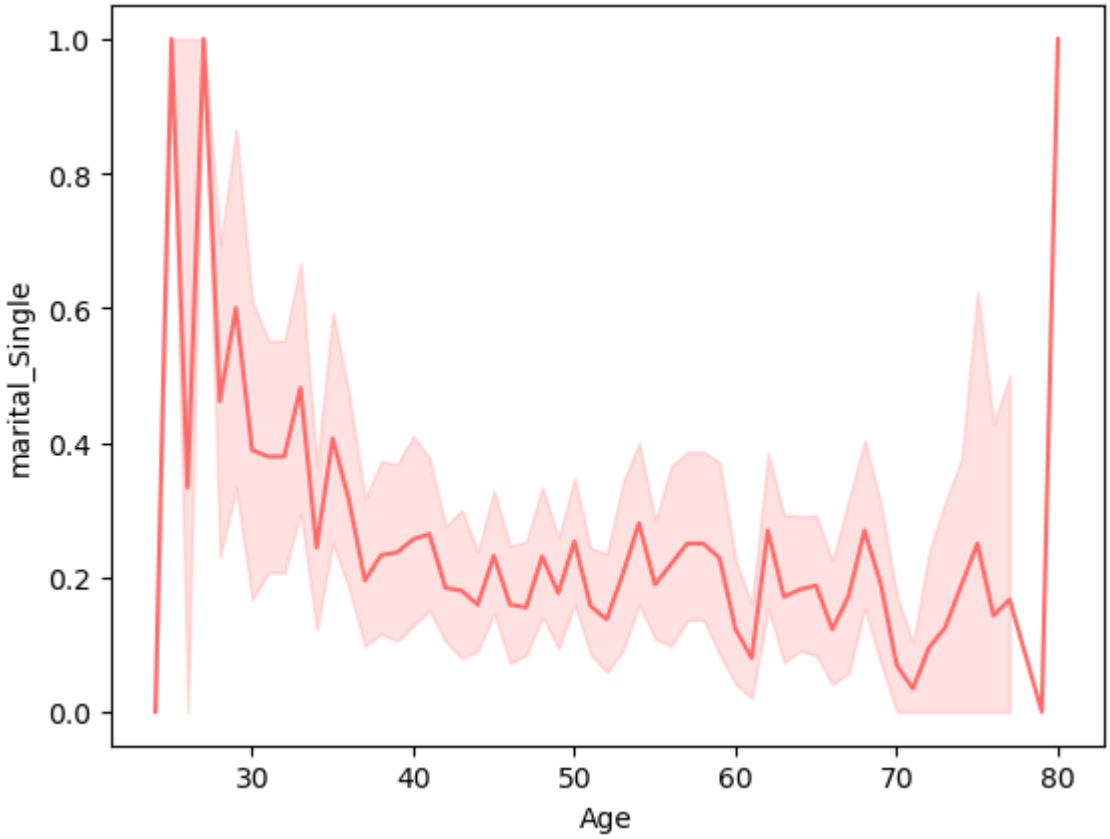
The relationship between the Single customers and the Single-status

In [263]:

```
# Create a line plot using seaborn to visualize the relationship between "Age" and "marita
l_Single"
sns.lineplot(x='Age', y='marital_Single',color='#FF6969', data=data)
```

Out[263]:

```
<Axes: xlabel='Age', ylabel='marital_Single'>
```



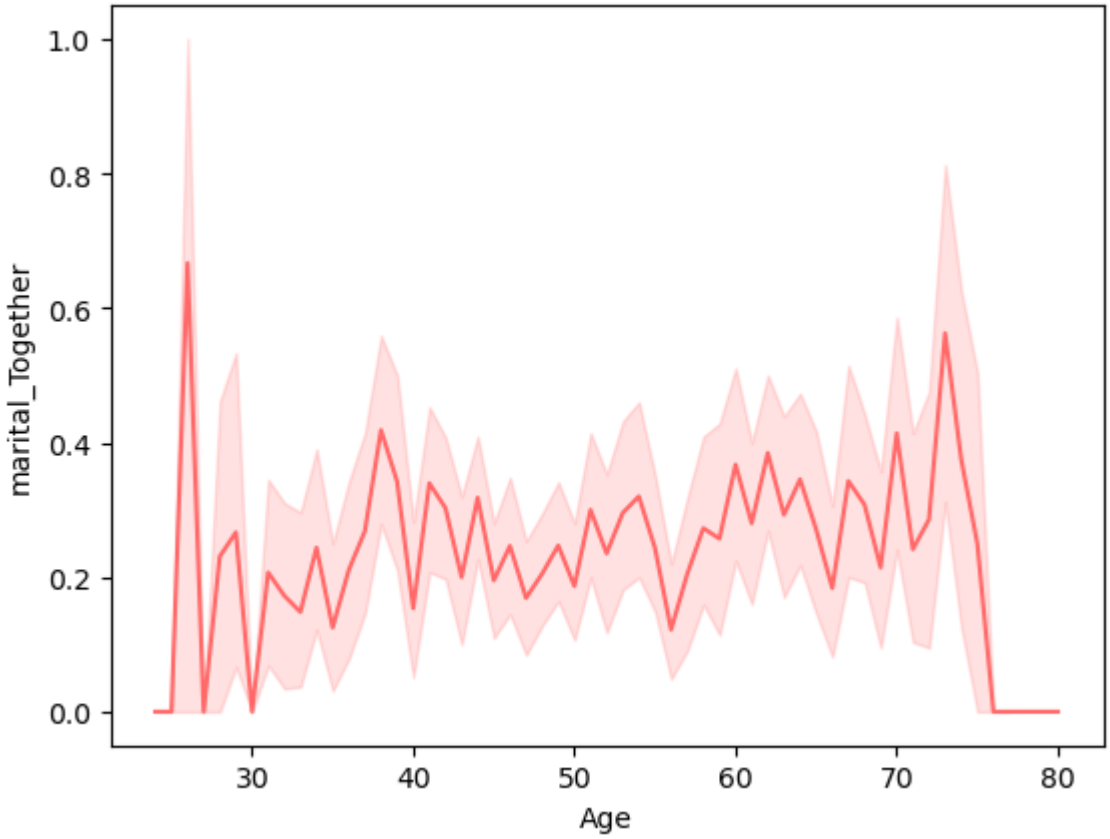
The relationship between the Divorced customers and the Together-status

In [264]:

```
# Create a line plot using seaborn to visualize the relationship between "Age" and "marita
l_Together"
sns.lineplot(x='Age', y='marital_Together',color='#FF6969', data=data)
```

Out[264]:

```
<Axes: xlabel='Age', ylabel='marital_Together'>
```



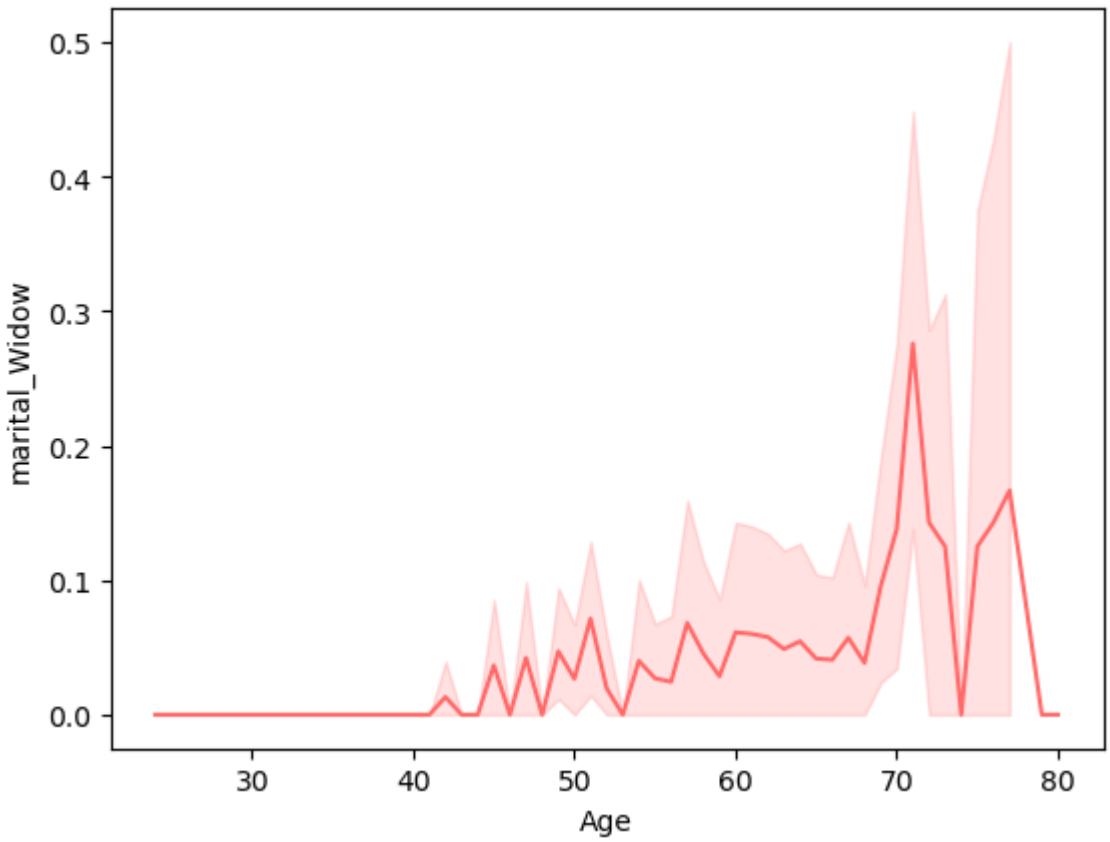
The relationship between the Divorced customers and the Widow-status

In [265]:

```
# Create a line plot using seaborn to visualize the relationship between "Age" and "marita
l_Widow"
sns.lineplot(x='Age', y='marital_Widow',color='#FF6969', data=data)
```

Out[265]:

```
<Axes: xlabel='Age', ylabel='marital_Widow'>
```



How many customer complaints in the last year?

In [266]:

```
data["Complain"].sum()
```

Out[266]:

```
20
```

Number of customers who filed a complaint in the last year is 20

In [267]:

```
# Number of marital_Discovered Customers
data["marital_Divorced"].sum()
```

Out[267]:

230

In [268]:

```
# Number of marital_Single Customers
data["marital_Single"].sum()
```

Out[268]:

477

In [269]:

```
# Number of marital_Widow Customers
data["marital_Widow"].sum()
```

Out[269]:

76

In [270]:

```
# Number of marital_Together Customers
data["marital_Together"].sum()
```

Out[270]:

568

In [271]:

```
# Number of marital_Married Customers
data["marital_Married"].sum()
```

Out[271]:

854

In [272]:

```
230/2205
```

Out[272]:

0.10430839002267574

In [273]:

```
477/2205
```

Out[273]:

```
0.2163265306122449
```

In [274]:

```
76/2205
```

Out[274]:

```
0.034467120181405894
```

In [275]:

```
568/2205
```

Out[275]:

```
0.2575963718820862
```

In [276]:

```
854/2205
```

Out[276]:

```
0.3873015873015873
```

The status of the company customers

10% from the company customers are Marital Divorced

22% from the company customers are Singles

3% from the company customers are Widows

26% from the company customers are Marital Together

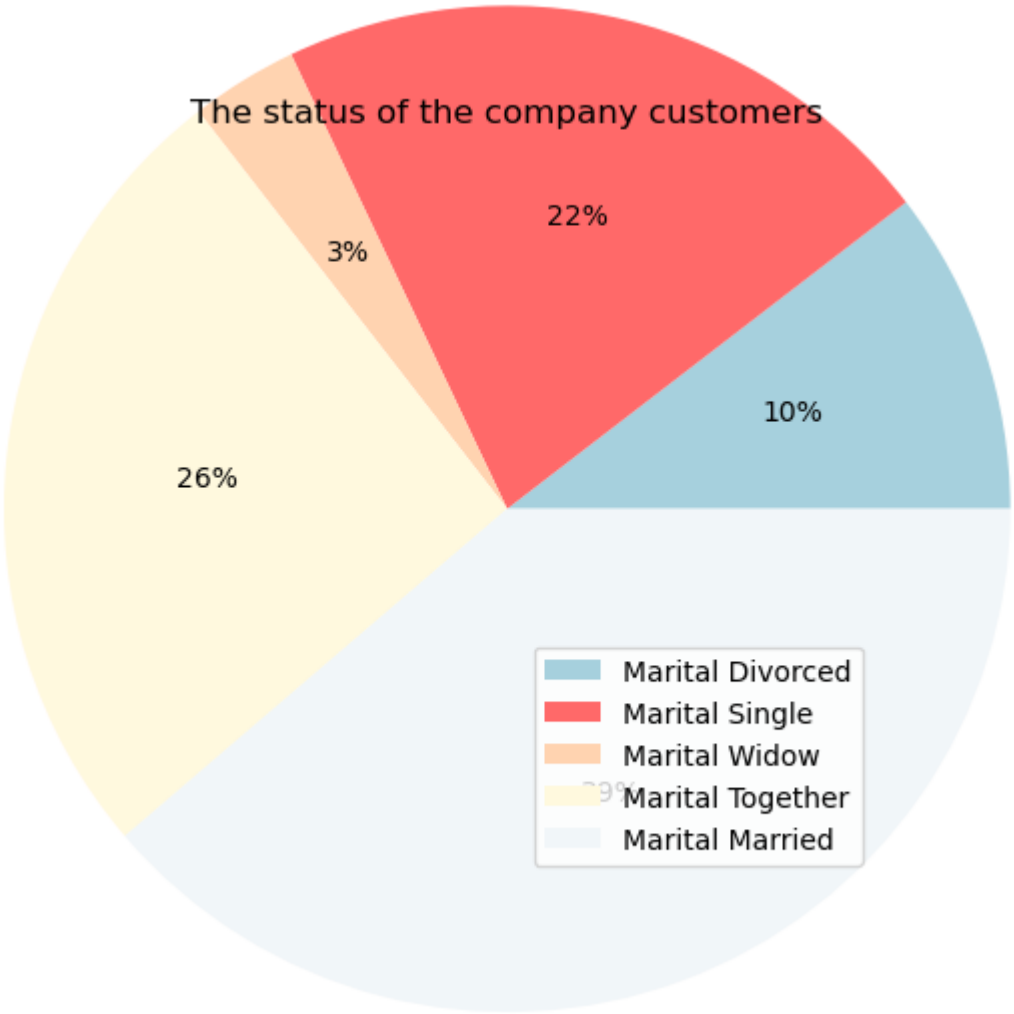
39% from the company customers are Married

In [277]:

```
# Visualize the percentage of customers based on their marital status using a pie chart
# Define the values for the pie chart
customers_status =[0.10430839002267574, 0.2163265306122449, 0.034467120181405894, 0.257596
3718820862, 0.3873015873015873]
# Create a pie chart with the specified values
plt.pie(customers_status,  radius=1.7, colors=["#A6D0DD","#FF6969", "#FFD3B0", "#FFF9DE",
"#F1F6F9"],
        rotatelabels=False, autopct= "%1.0f%%")
# Add a title to the plot
plt.title('The status of the company customers')
# Add a legend to the plot with the specified labels
plt.legend(labels=["Marital Divorced", "Marital Single", "Marital Widow", "Marital Togethe
r", "Marital Married"])
```

Out[277]:

```
<matplotlib.legend.Legend at 0x7800059187c0>
```



The relationship between Marital-Divorced status and the number of days since the last purchase

In [278]:

```
# Create a 2D kernel density plot using seaborn to visualize the relationship between "mar
ital_Divorced" and "Recency"
sns.kdeplot(data=data, x='marital_Divorced', y='Recency',shade=True)
```

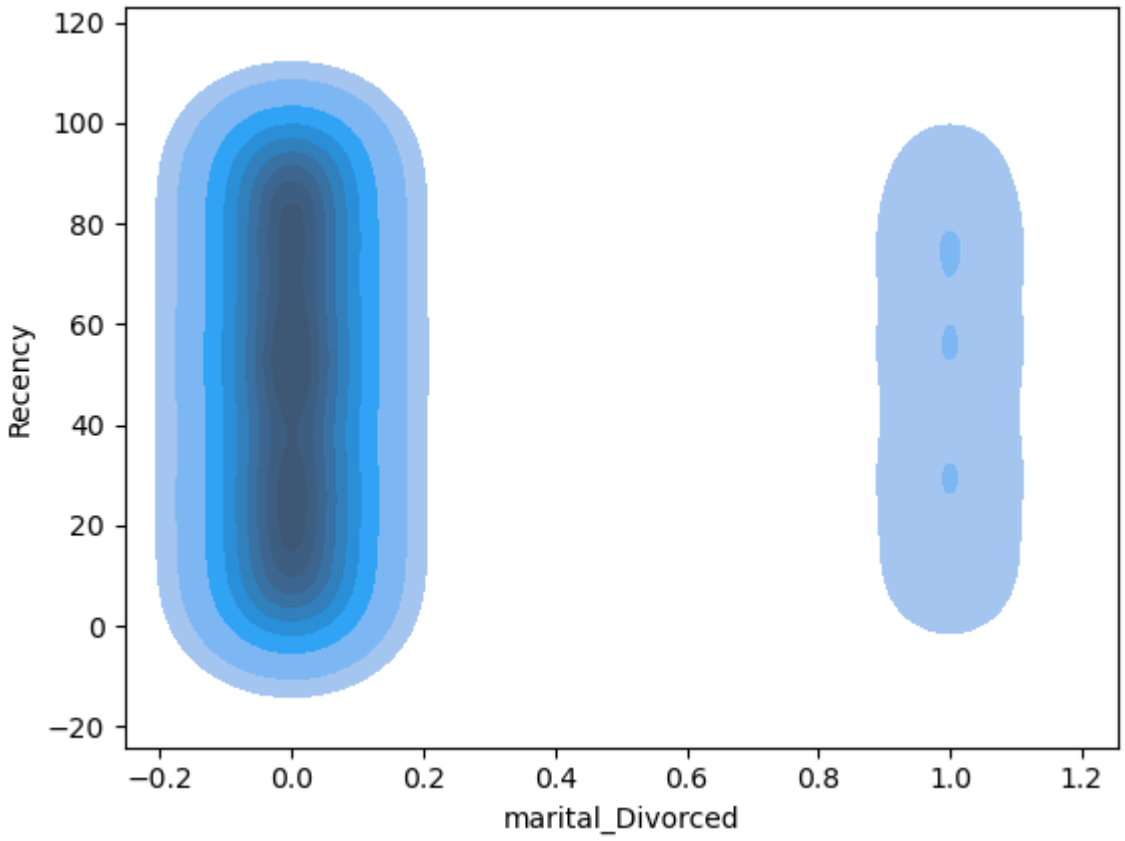
```
/tmp/ipykernel_20/1022136289.py:2: FutureWarning:

`shade` is now deprecated in favor of `fill`; setting `fill=True`.
This will become an error in seaborn v0.14.0; please update your code.

  sns.kdeplot(data=data, x='marital_Divorced', y='Recency',shade=True)
```

Out[278]:

```
<Axes: xlabel='marital_Divorced', ylabel='Recency'>
```



The relationship between Marital-Single status and the number of days since the last purchase

In [279]:

```
# Create a 2D kernel density plot using seaborn to visualize the relationship between "mar
ital_Single" and "Recency"
sns.kdeplot(data=data, x='marital_Single', y='Recency',shade=True)
```

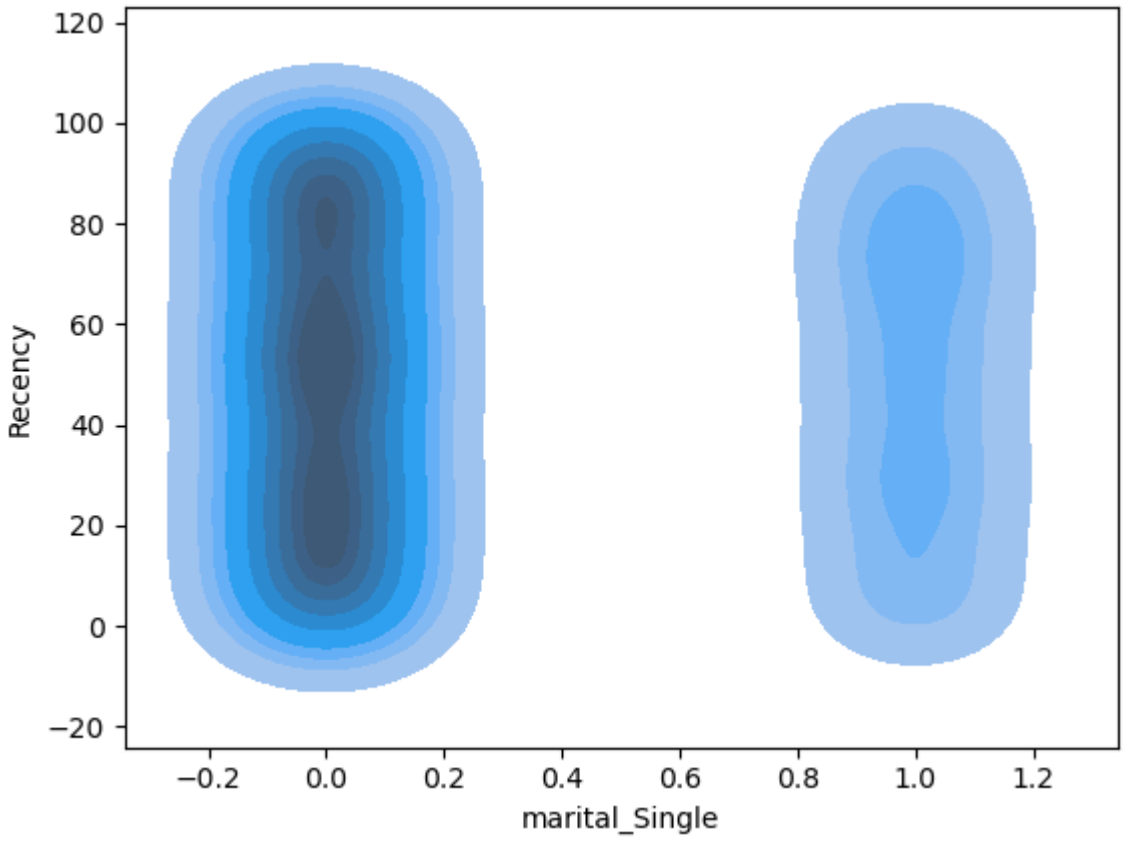
```
/tmp/ipykernel_20/1889469451.py:2: FutureWarning:

`shade` is now deprecated in favor of `fill`; setting `fill=True`.
This will become an error in seaborn v0.14.0; please update your code.

  sns.kdeplot(data=data, x='marital_Single', y='Recency',shade=True)
```

Out[279]:

```
<Axes: xlabel='marital_Single', ylabel='Recency'>
```



The relationship between Marital-Widow status and the number of days since the last purchase

In [280]:

```
# Create a 2D kernel density plot using seaborn to visualize the relationship between "mar
ital_Widow" and "Recency"
sns.kdeplot(data=data, x='marital_Widow', y='Recency',shade=True)
```

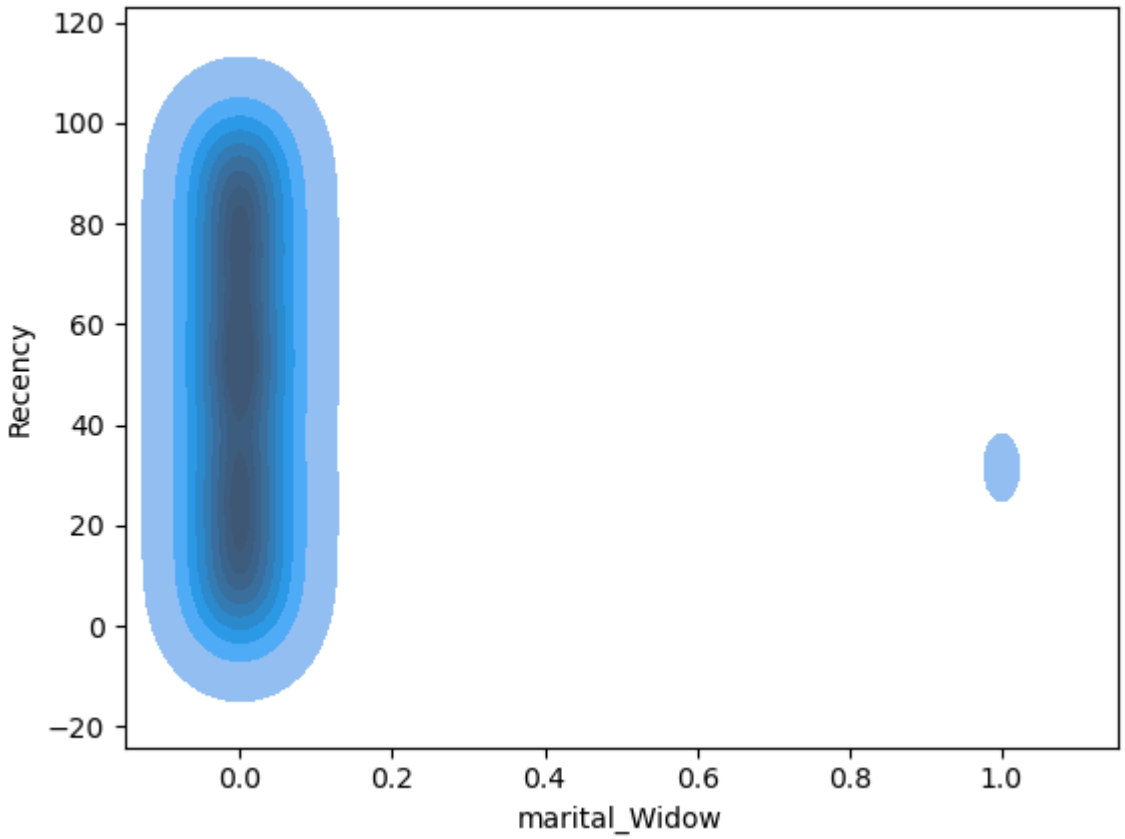
```
/tmp/ipykernel_20/3125291054.py:2: FutureWarning:

`shade` is now deprecated in favor of `fill`; setting `fill=True`.
This will become an error in seaborn v0.14.0; please update your code.

  sns.kdeplot(data=data, x='marital_Widow', y='Recency',shade=True)
```

Out[280]:

```
<Axes: xlabel='marital_Widow', ylabel='Recency'>
```



The relationship between Marital-Married status and the number of days since the last purchase

In [281]:

```
# Create a 2D kernel density plot using seaborn to visualize the relationship between "mar
ital_Married" and "Recency"
sns.kdeplot(data=data, x='marital_Married', y='Recency',shade=True)
```

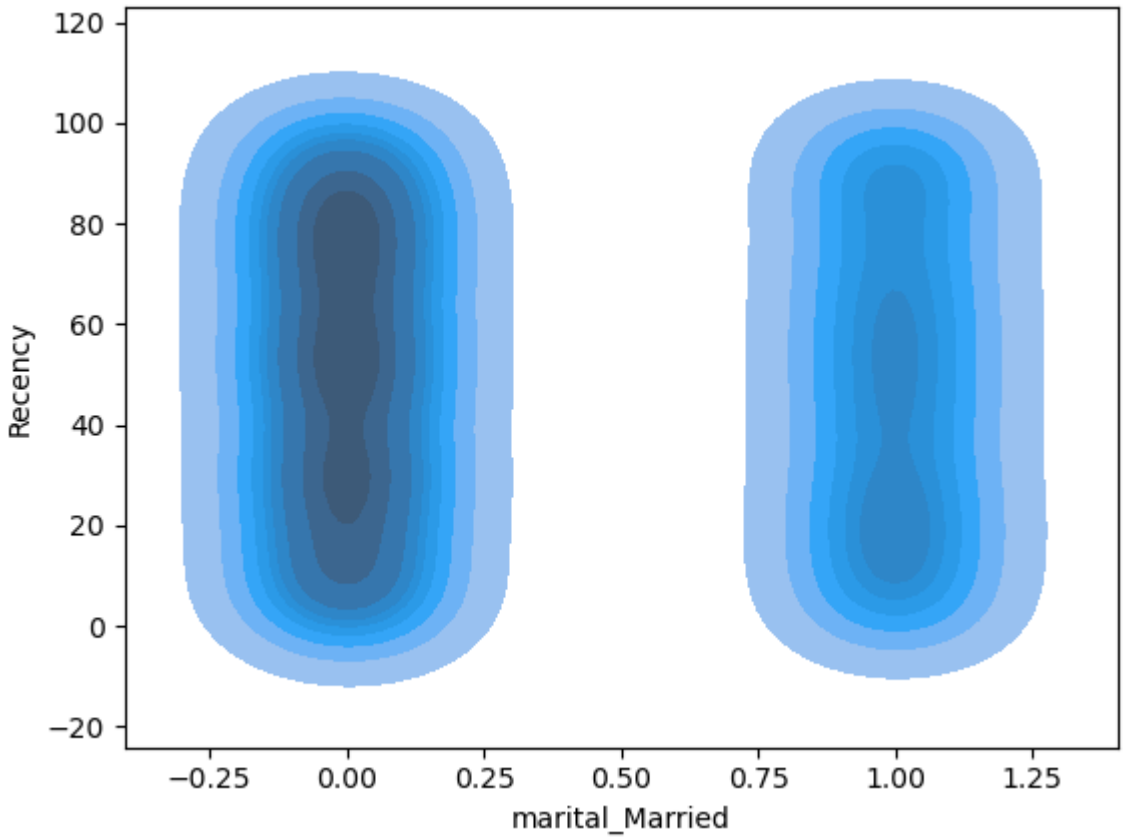
```
/tmp/ipykernel_20/341482823.py:2: FutureWarning:

`shade` is now deprecated in favor of `fill`; setting `fill=True`.
This will become an error in seaborn v0.14.0; please update your code.

  sns.kdeplot(data=data, x='marital_Married', y='Recency',shade=True)
```

Out[281]:

```
<Axes: xlabel='marital_Married', ylabel='Recency'>
```



The relationship between Marital-Together status and the number of days since the last purchase

In [282]:

```
# Create a 2D kernel density plot using seaborn to visualize the relationship between "mar
ital_Together" and "Recency"
sns.kdeplot(data=data, x='marital_Together', y='Recency',shade=True)
```

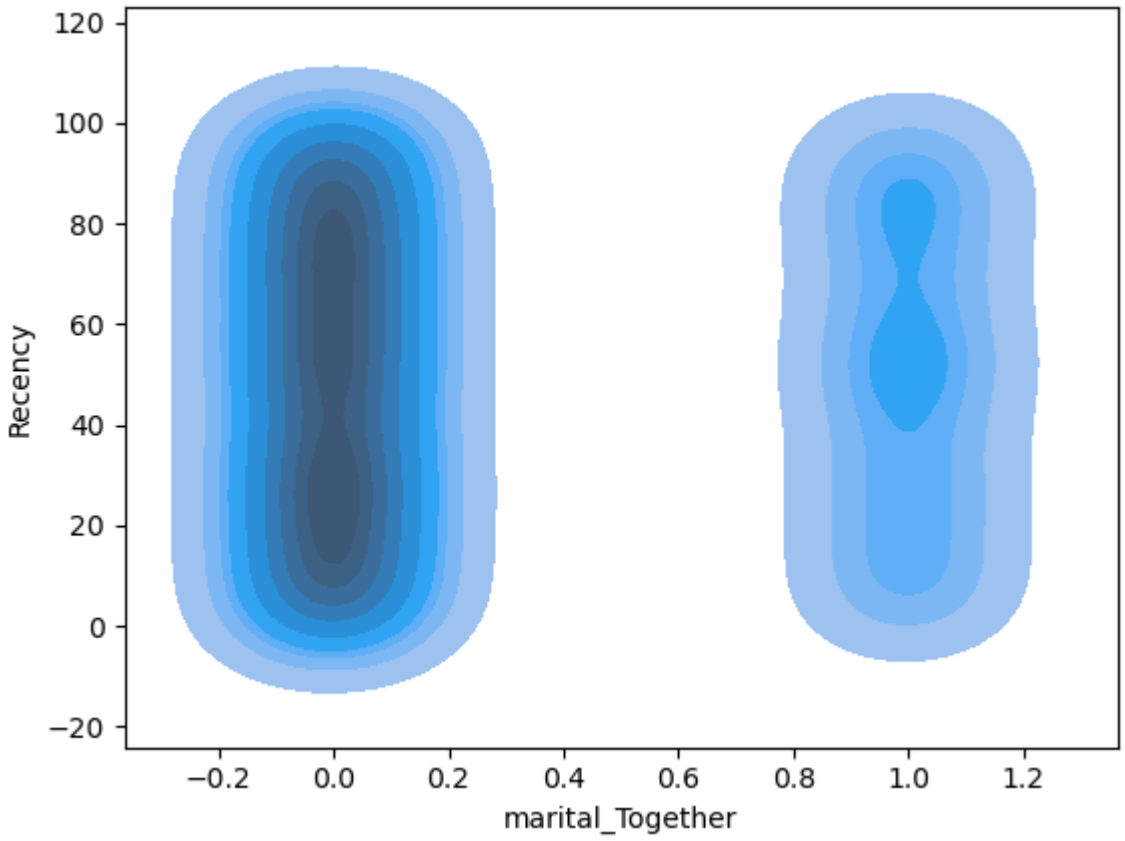
```
/tmp/ipykernel_20/3097359276.py:2: FutureWarning:

`shade` is now deprecated in favor of `fill`; setting `fill=True`.
This will become an error in seaborn v0.14.0; please update your code.

  sns.kdeplot(data=data, x='marital_Together', y='Recency',shade=True)
```

Out[282]:

```
<Axes: xlabel='marital_Together', ylabel='Recency'>
```



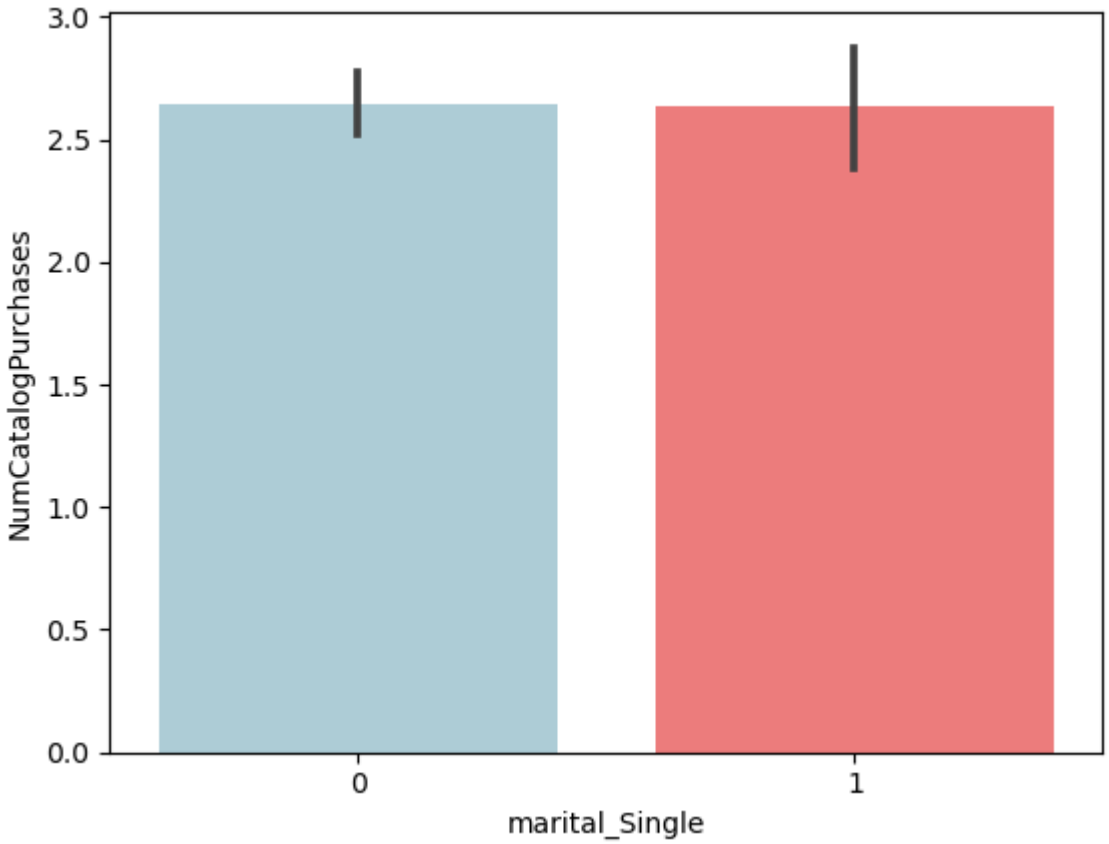
What is the average purchase of Single from company through the website, store or catalog?

In [283]:

```
# Create a bar plot using seaborn to visualize the relationship between "marital_Single" a
nd "NumCatalogPurchases"
color =["#A6D0DD","#FF6969"]
sns.barplot(x="marital_Single", y="NumCatalogPurchases", data=data, palette=color)
```

Out[283]:

```
<Axes: xlabel='marital_Single', ylabel='NumCatalogPurchases'>
```

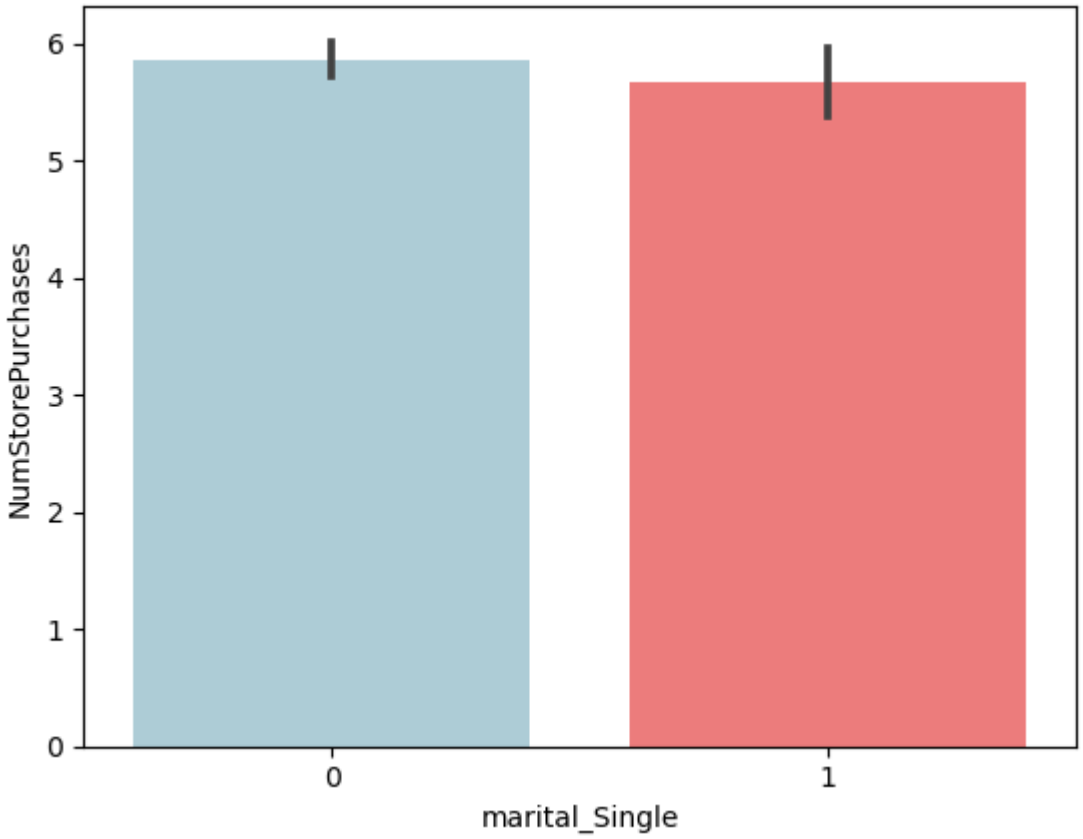


In [284]:

```
# Create a bar plot using seaborn to visualize the relationship between "marital_Single" a
nd "NumStorePurchases"
color =["#A6D0DD","#FF6969"]
sns.barplot(x="marital_Single", y="NumStorePurchases", data=data, palette=color)
```

Out[284]:

```
<Axes: xlabel='marital_Single', ylabel='NumStorePurchases'>
```

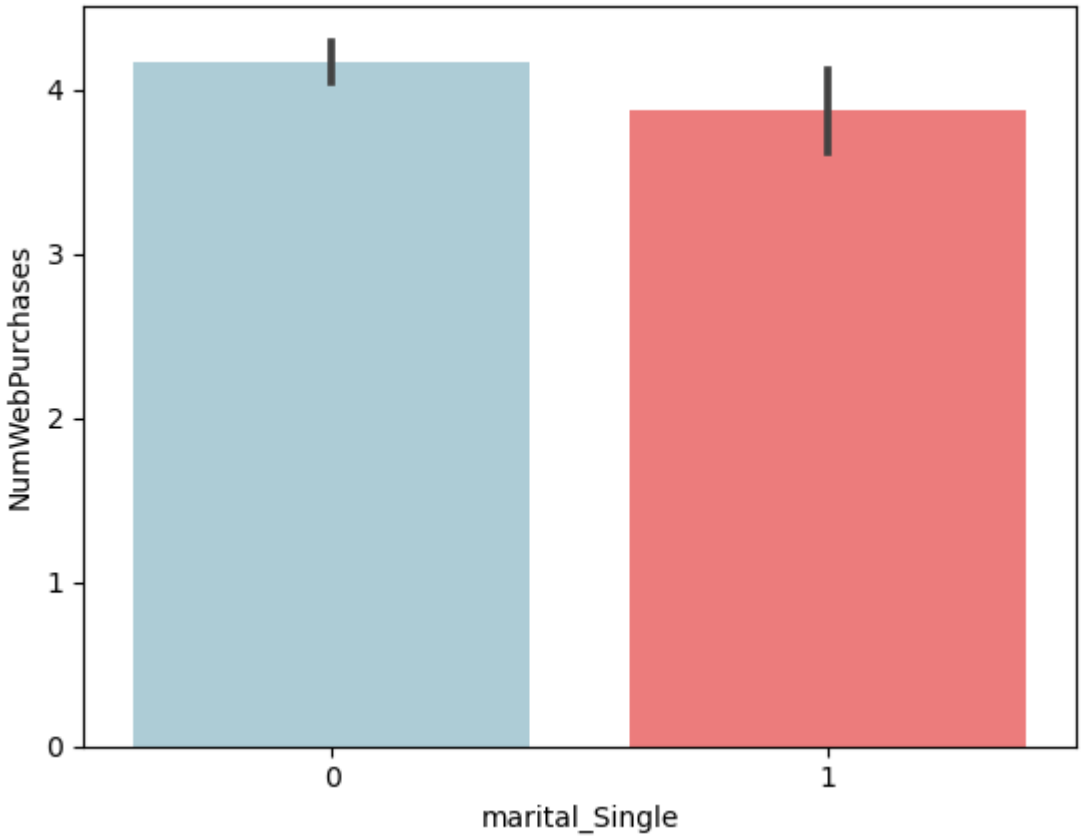


In [285]:

```
# Create a bar plot using seaborn to visualize the relationship between "marital_Single" a
nd "NumWebPurchases"
color =["#A6D0DD","#FF6969"]
sns.barplot(x="marital_Single", y="NumWebPurchases", data=data, palette=color)
```

Out[285]:

```
<Axes: xlabel='marital_Single', ylabel='NumWebPurchases'>
```



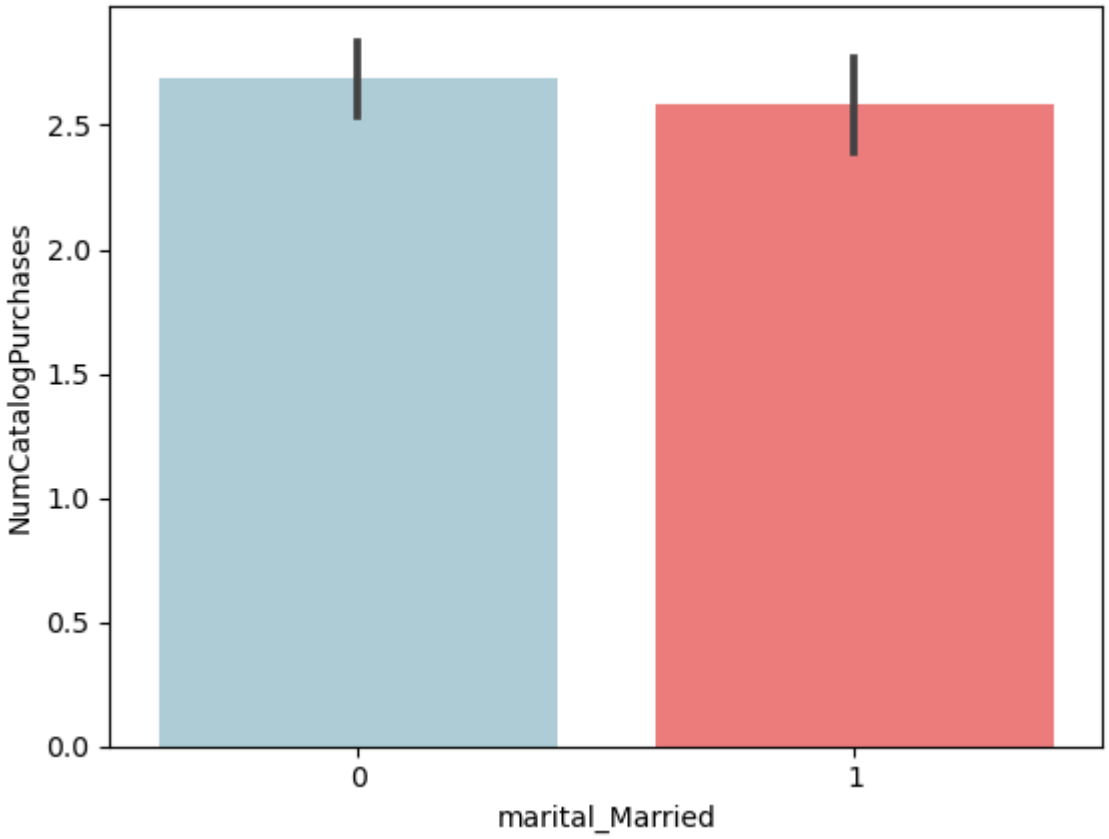
What is the average purchase of Married from company through the website, store or catalog?

In [286]:

```
# Create a bar plot using seaborn to visualize the relationship between "marital_Married"
and "NumCatalogPurchases"
color =["#A6D0DD","#FF6969"]
sns.barplot(x="marital_Married", y="NumCatalogPurchases", data=data, palette=color)
```

Out[286]:

```
<Axes: xlabel='marital_Married', ylabel='NumCatalogPurchases'>
```

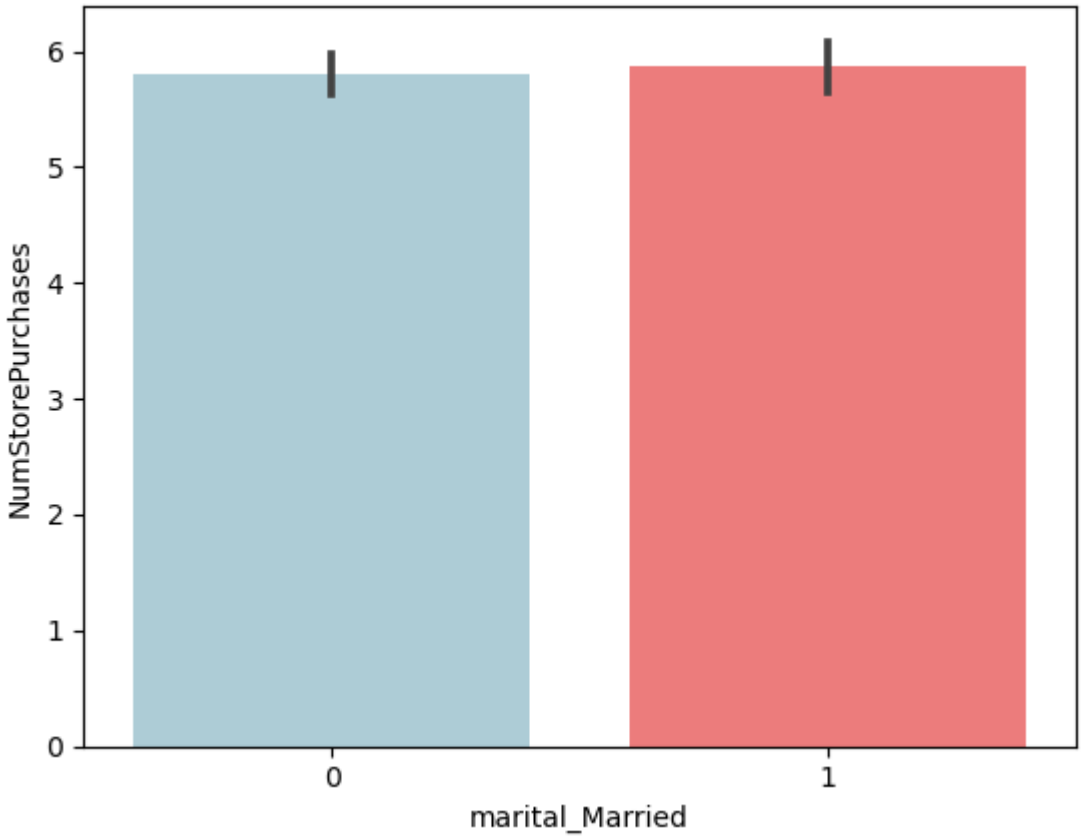


In [287]:

```
# Create a bar plot using seaborn to visualize the relationship between "marital_Married"
and "NumStorePurchases"
color =["#A6D0DD","#FF6969"]
sns.barplot(x="marital_Married", y="NumStorePurchases", data=data, palette=color)
```

Out[287]:

```
<Axes: xlabel='marital_Married', ylabel='NumStorePurchases'>
```

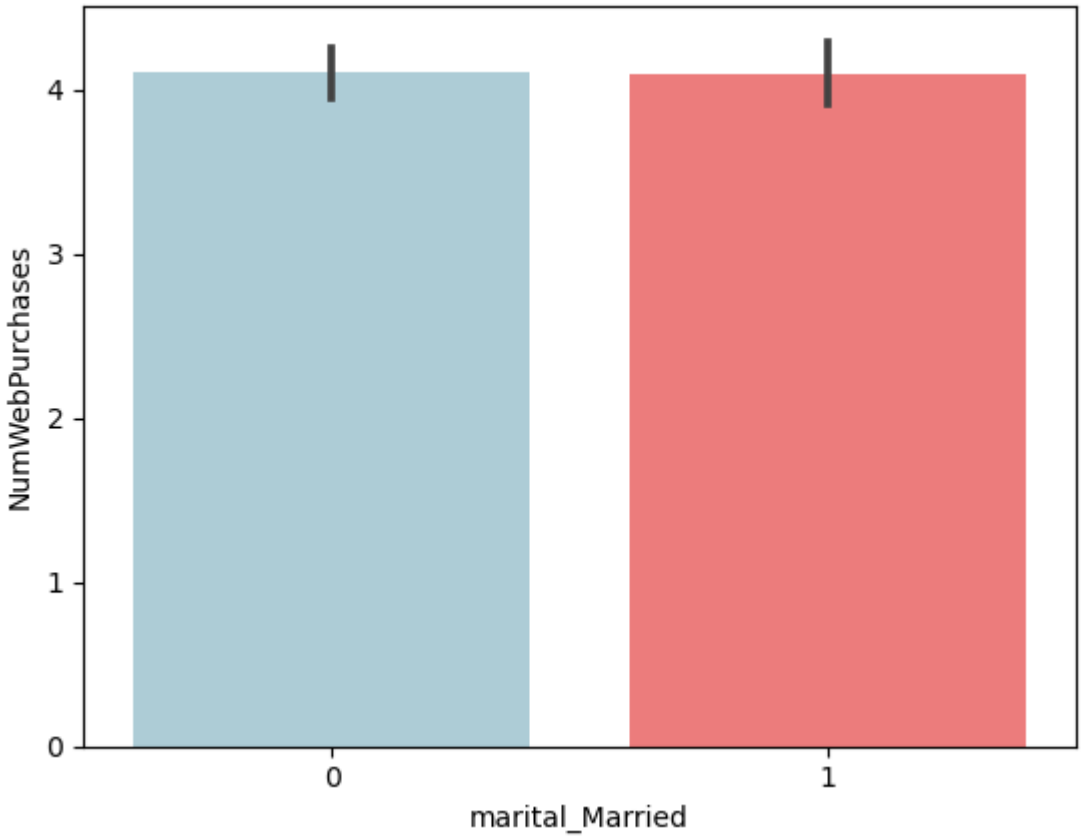


In [288]:

```
# Create a bar plot using seaborn to visualize the relationship between "marital_Married"
and "NumWebPurchases"
color =["#A6D0DD","#FF6969"]
sns.barplot(x="marital_Married", y="NumWebPurchases", data=data, palette=color)
```

Out[288]:

```
<Axes: xlabel='marital_Married', ylabel='NumWebPurchases'>
```



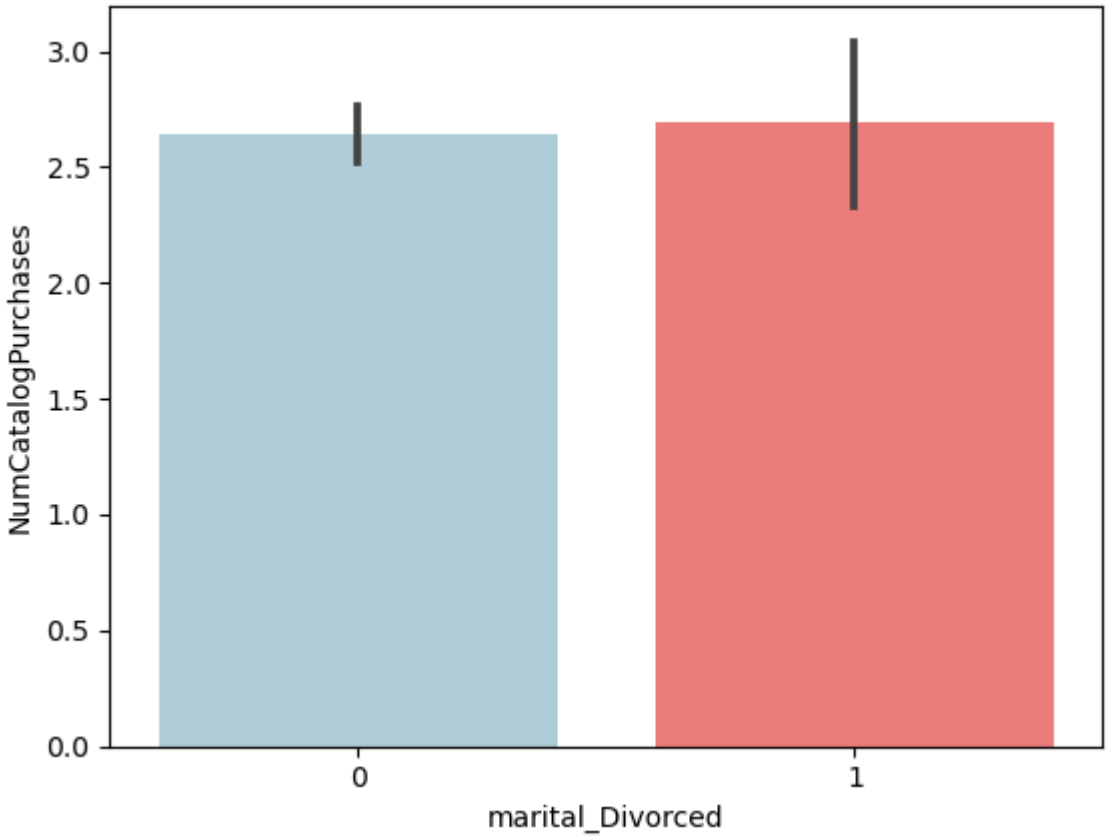
What is the average purchase of Divorced from company through the website, store or catalog?

In [289]:

```
# Create a bar plot using seaborn to visualize the relationship between "marital_Divorced"
and "NumCatalogPurchases"
color =["#A6D0DD","#FF6969"]
sns.barplot(x="marital_Divorced", y="NumCatalogPurchases", data=data, palette=color)
```

Out[289]:

```
<Axes: xlabel='marital_Divorced', ylabel='NumCatalogPurchases'>
```

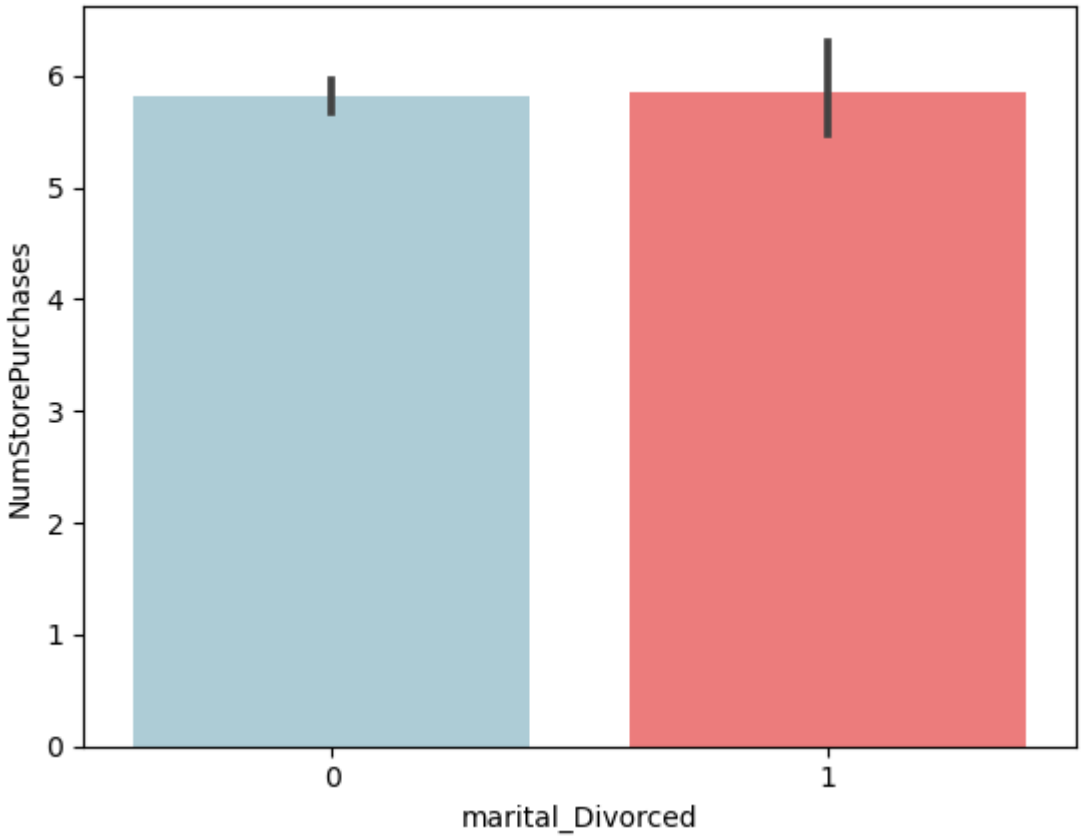


In [290]:

```
# Create a bar plot using seaborn to visualize the relationship between "marital_Divorced"
and "NumStorePurchases"
color =["#A6D0DD","#FF6969"]
sns.barplot(x="marital_Divorced", y="NumStorePurchases", data=data, palette=color)
```

Out[290]:

```
<Axes: xlabel='marital_Divorced', ylabel='NumStorePurchases'>
```

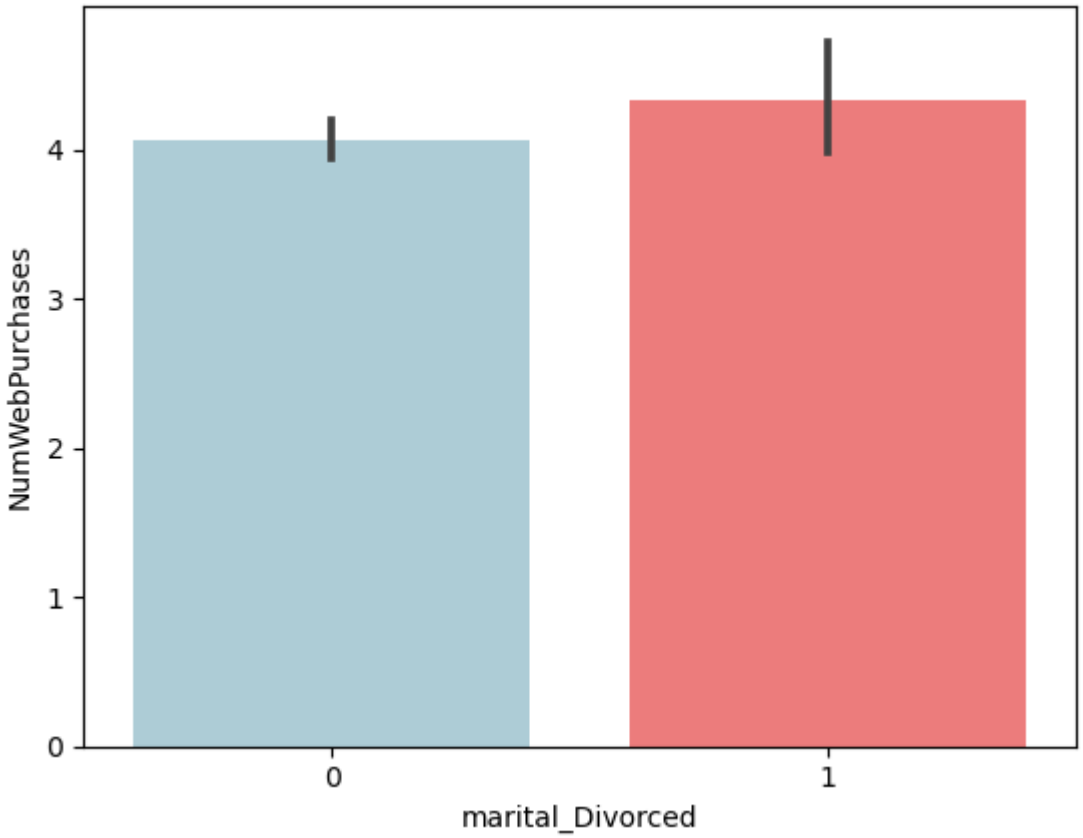


In [291]:

```
# Create a bar plot using seaborn to visualize the relationship between "marital_Divorced"
and "NumWebPurchases"
color =["#A6D0DD","#FF6969"]
sns.barplot(x="marital_Divorced", y="NumWebPurchases", data=data, palette=color)
```

Out[291]:

```
<Axes: xlabel='marital_Divorced', ylabel='NumWebPurchases'>
```



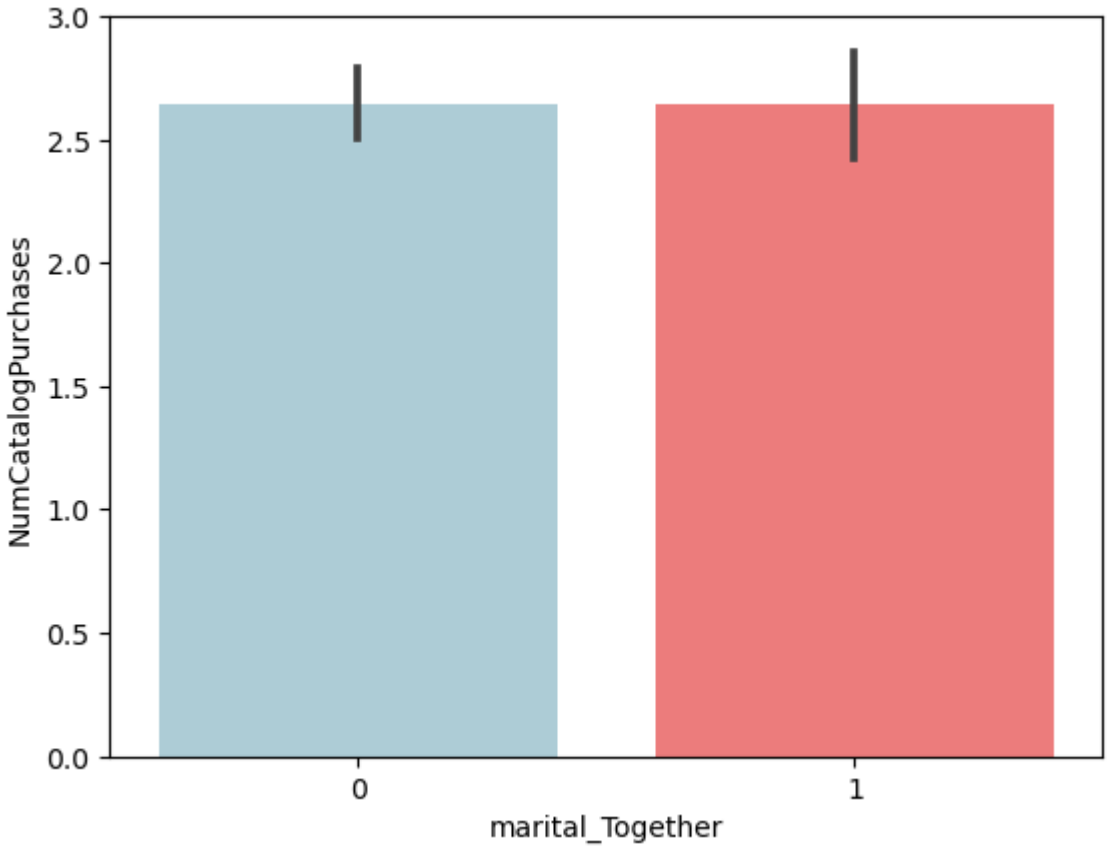
What is the average purchase of marital_Together from company through the website, store or catalog?

In [292]:

```
# Create a bar plot using seaborn to visualize the relationship between "marital_Together"
and "NumCatalogPurchases"
color =["#A6D0DD","#FF6969"]
sns.barplot(x="marital_Together", y="NumCatalogPurchases", data=data, palette=color)
```

Out[292]:

```
<Axes: xlabel='marital_Together', ylabel='NumCatalogPurchases'>
```

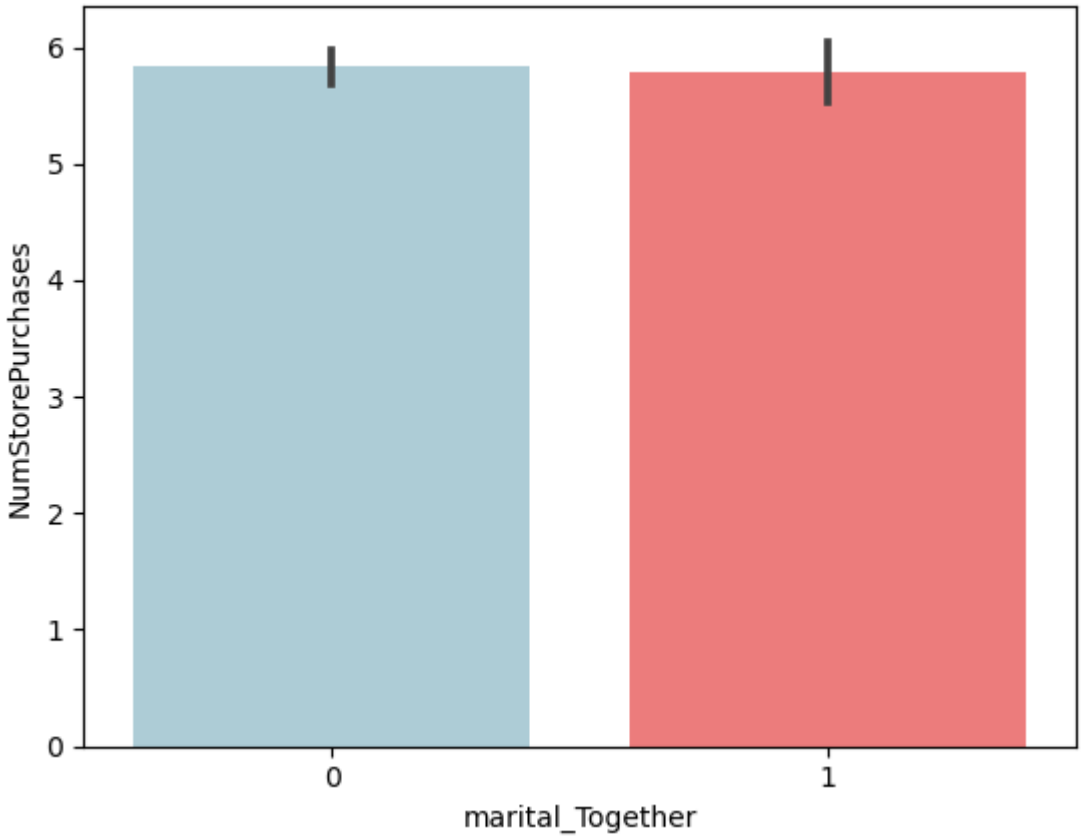


In [293]:

```
# Create a bar plot using seaborn to visualize the relationship between "marital_Together"
and "NumStorePurchases"
color =["#A6D0DD","#FF6969"]
sns.barplot(x="marital_Together", y="NumStorePurchases", data=data, palette=color)
```

Out[293]:

```
<Axes: xlabel='marital_Together', ylabel='NumStorePurchases'>
```

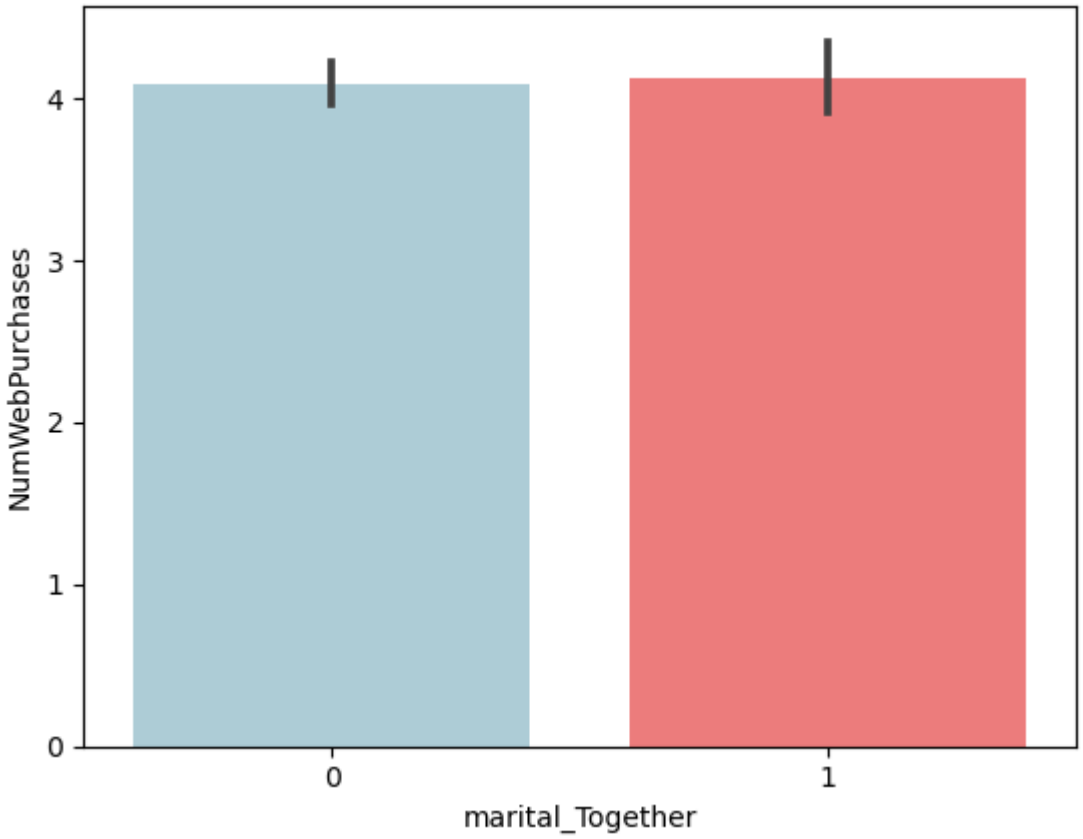


In [294]:

```
# Create a bar plot using seaborn to visualize the relationship between "marital_Together"
and "NumWebPurchases"
color =["#A6D0DD","#FF6969"]
sns.barplot(x="marital_Together", y="NumWebPurchases", data=data, palette=color)
```

Out[294]:

```
<Axes: xlabel='marital_Together', ylabel='NumWebPurchases'>
```



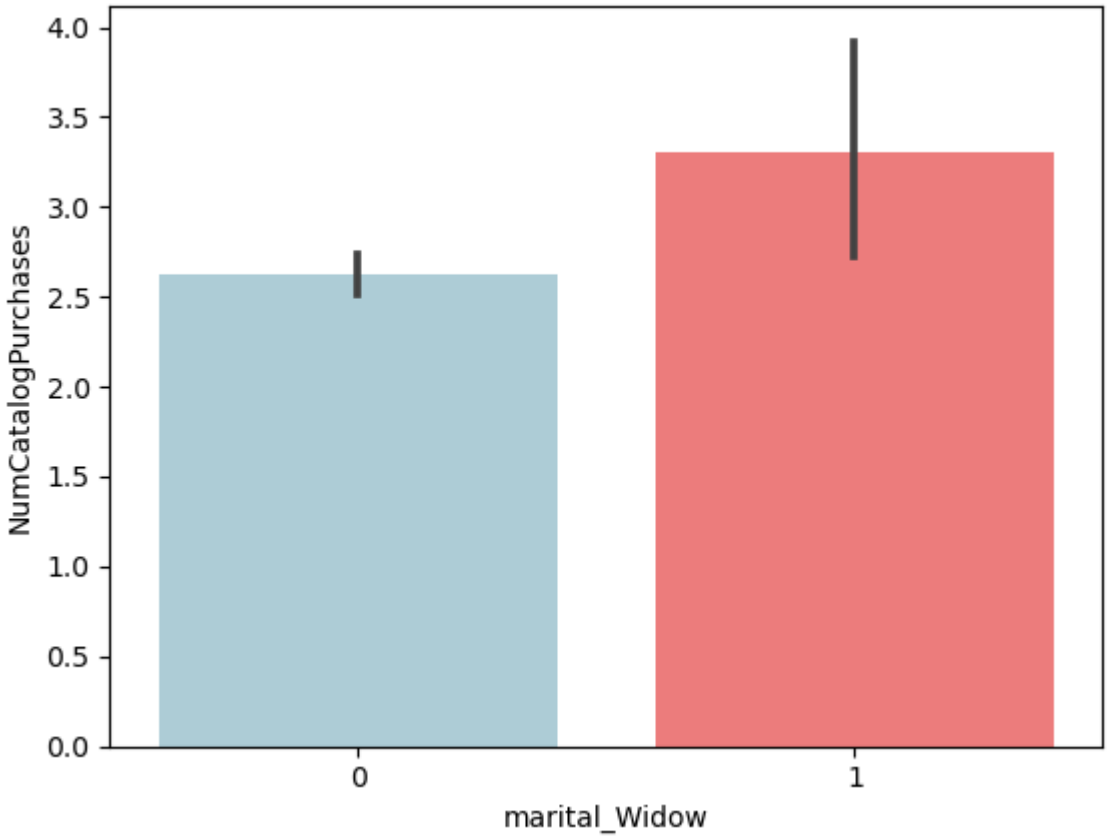
What is the average purchase of Widow from company through the website, store or catalog?

In [295]:

```
# Create a bar plot using seaborn to visualize the relationship between "marital_Widow" an
d "NumCatalogPurchases"
color =["#A6D0DD","#FF6969"]
sns.barplot(x="marital_Widow", y="NumCatalogPurchases", data=data, palette=color)
```

Out[295]:

```
<Axes: xlabel='marital_Widow', ylabel='NumCatalogPurchases'>
```

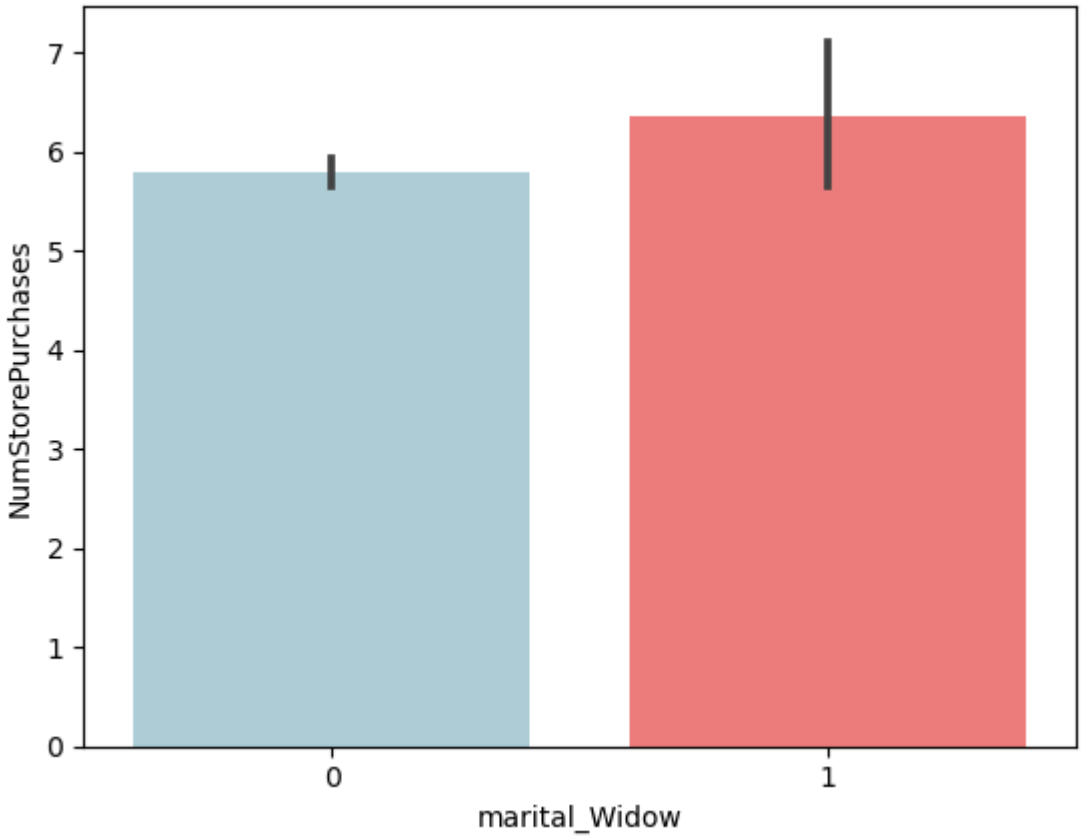


In [296]:

```
# Create a bar plot using seaborn to visualize the relationship between "marital_Widow" an
d "NumStorePurchases"
color =["#A6D0DD","#FF6969"]
sns.barplot(x="marital_Widow", y="NumStorePurchases", data=data, palette=color)
```

Out[296]:

```
<Axes: xlabel='marital_Widow', ylabel='NumStorePurchases'>
```

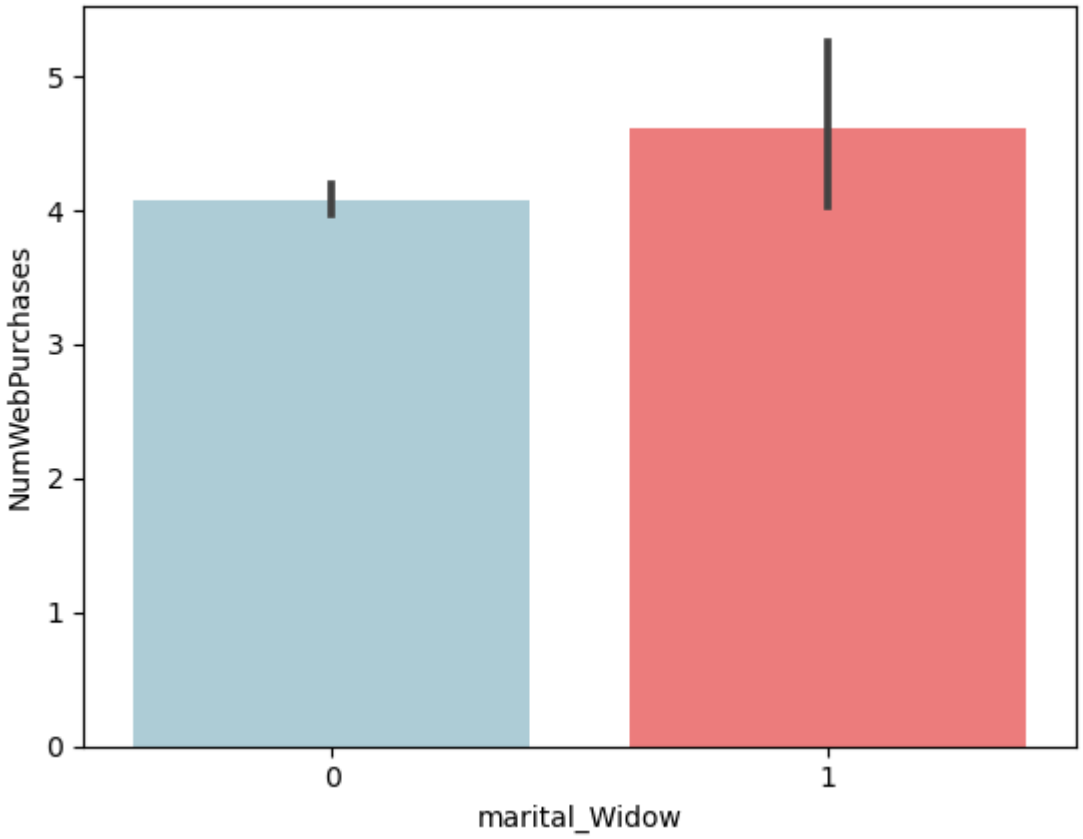


In [297]:

```
# Create a bar plot using seaborn to visualize the relationship between "marital_Widow" an
d "NumWebPurchases"
color =["#A6D0DD","#FF6969"]
sns.barplot(x="marital_Widow", y="NumWebPurchases", data=data, palette=color)
```

Out[297]:

```
<Axes: xlabel='marital_Widow', ylabel='NumWebPurchases'>
```



In [ ]:

In [ ]:

In [ ]: