In [1]:

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

```
D:\rgzn\lib\site-packages\numpy\_distributor_init.py:32: UserWarning: loaded more th
an 1 DLL from .libs:
D:\rgzn\lib\site-packages\numpy\.libs\libopenblas.NOIJJG62EMASZI6NYURL6JBKM4EVBGM7.g
fortran-win_amd64.dll
D:\rgzn\lib\site-packages\numpy\.libs\libopenblas.XWYDX2IKJW2NMTWSFYNGFUWKQU3LYTCZ.g
fortran-win_amd64.dll
  stacklevel=1)
```

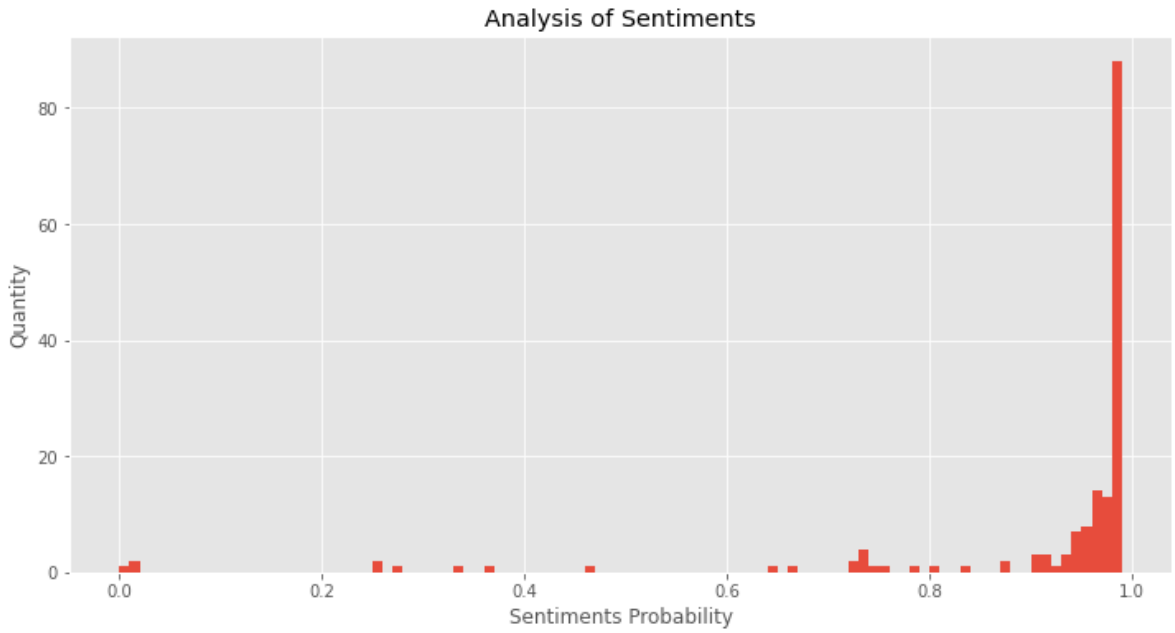
In [2]:

```
df1 = pd.read_excel('./data/clean_data.xlsx')
```

根据情感分析去统计内容的情感分布，得出大部分文章内容集中在0.8-1之间，说明大多数都是好评为主

In [27]:

```
plt.style.use('ggplot')
plt.figure(figsize=(12,6))
plt.hist(df1['情感分值'], bins = np.arange(0, 1, 0.01), facecolor = '#E74C3C')
plt.xlabel('Sentiments Probability')
plt.ylabel('Quantity')
plt.title('Analysis of Sentiments')
plt.savefig('Analysis of Sentiments.jpg')
plt.show()
```



In [3]:

```
data1 = pd.read_excel('./data/去哪儿游记.xlsx')
content1 = data1['正文']
data2 = pd.read_excel('./data/携程游记.xlsx')
content2 = data2['正文']
data3 = pd.read_excel('./data/马蜂窝游记.xlsx')
content3 = data3['正文']
data4 = pd.read_excel('./data/驴妈妈游记.xlsx').iloc[:11]
data4['正文'] = data4['游记内容']
content4 = data4['正文']
```

In [4]:

```
data1.head()
```

Out[4]:

	页面网址	路线	用户id	日记id	发表时间	浏览量	出发日期	逗留天数
0	http://travel.qunar.com/travelbook/note/7553907	前言;\r\n说说这次旅行;\r\n序：初见三亚，暴走之旅;\r\n三亚;\r\n美图预告：...	吴人少白	北纬18度的浪漫，闺蜜团三亚暴走之旅！【附自由行攻略】	2020-12-08	1.7万	2020/07/18	7
1	http://travel.qunar.com/travelbook/note/7666352	出行前：;\r\n第1天4.6;\r\n美兰国际机场;\r\n第2天4.7;\r\n万宁石梅...	小歪	🛩海口自驾🚘万宁-陵水-三亚ლ亲子游	2021-04-07	6875	2021/04/06	8
2	http://travel.qunar.com/travelbook/note/7630336	说说这次旅行;\r\n行程路线;\r\n住宿;\r\n行程花费;\r\n第1天飞抵海岛，去领...	拖沓天王1	候鸟来袭前的魅力海岛游	2020-11-09	1.2万	2020/10/30	8
3	http://travel.qunar.com/travelbook/note/7618202	请到天涯海角来，这里四季春常在;\r\n三亚纪忆的那些美好瞬间;\r\n6天5晚三亚经典自由...	驴说旅行	余夏金秋去三亚：6天5晚“玩在当地，吃在当季”的三亚纪忆·夏日食光	2020-09-05	1.4万	2020/08/27	6

	页面网址	路线	用户id	日记id	发表时间	浏览量	出发日期	逗留天数
4	http://travel.qunar.com/travelbook/note/7607157	序言;\r\n说说这次旅行;\r\n实用性（行程安排、准备）;\r\n照片秀一波;\r\n第...	冰鉴初心	KK寻那抹“蓝”，一路向南	2020-08-03	7881	2020/08/04	1

In [5]:

```
data2.head()
```

Out[5]:

	页面网址	发表时间	标题	作者	城市	天数	旅游时间	人均
0	https://you.ctrip.com/travels/sanya61/4014427....	2021-06-01 20:32	三亚美食攻略|住进奢华亚特兰斯蒂，赏三亚各种美食	vivi慢生活	三亚	6天	5月	15000元
1	https://you.ctrip.com/travels/sanya61/3997496....	2021-03-01 00:49	三亚怎么玩?这份吃喝玩乐宝藏攻略，你可得收好了	小飞侠Finn	三亚	4天	3月	2000元

	页面网址	发表时间	标题	作者	城市	天数	旅游时间	人均
2	https://you.ctrip.com/travels/sanya61/4014427....	2021-05-22 10:01	三亚只有海？你OUT啦，快收下这份独特的游玩秘籍	滢萱	三亚	3天	5月	3000元
3	https://you.ctrip.com/travels/sanya61/3973888....	2020-09-29 23:16	泡酒店、吃美食、沙滩玩耍，三亚亲子游的正确打开方式！	敏华爱美丽	三亚	7天	9月	2500元

	页面网址	发表时间	标题	作者	城市	天数	旅游时间	人均
4	https://you.ctrip.com/travels/hainan100001/396...	2020.08.05	孕期旅行｜一个身体两颗心跳，带你去天之涯海之角，邂逅网红却又安静的三亚	余頭小姐_Rachel	海南	6天	7月	2500元

In [6]:

```
data3.head()
```

Out[6]:

	页面网址	标题	作者	作者所在地	发布时间	浏览数	评论数	赞同数	收藏数	分享数	.
0	http://www.mafengwo.cn/i/19825537.html	今年，一定要去海南自驾“薅羊毛”啊！	余小傻	杭州	2020-06-11 18:18:00	1.8w	39.0	NaN	896	4	.
1	http://www.mafengwo.cn/i/20012791.html	疫情海南行202006	欧歌	中国	2020-06-25 15:50:00	771	0.0	NaN	3	0	.
2	http://www.mafengwo.cn/i/20803264.html	三亚，亚龙湾-亚特兰蒂斯4日游，带娃带老妈的进～呕血心得	猫儿	上海	2020-09-10 17:11:00	1407	0.0	NaN	12	0	.
3	http://www.mafengwo.cn/i/19883041.html	初夏最美城市周边游：三亚网红景点+美食攻略	践行旅志	广州	2020-06-16 10:53:00	4.4w	37.0	NaN	158	101	.
4	http://www.mafengwo.cn/i/21467806.html	1、海南，在最热的季节来晒太阳，最后悔的是没下海游泳！	黑白素心笺	深圳	2021-02-03 03:07:00	2816	50.0	NaN	39	2	.

5 rows × 37 columns

In [7]:

```
data4.head()
```

Out[7]:

	页面网址	发布人名称	发布时间	出游时间	照片数	游记标题	章节目录	游记内容	正文
0	http://www.lvmama.com/trip/show/884828	dinhoidinh	2021-04-19	12月出游	139张照片	赴一场与海的约定丨在海南提前过夏天	章节目录\n\n...	前言\n\n\n...	前言\n\n\n...
1	http://www.lvmama.com/trip/show/884810	被遗忘的时光-	2021-04-17	12月出游	186张照片	海南·我等不了清晨的日出，但我可以慵懒的等日落	章节目录\n\n...	前言\n\n\n...	前言\n\n\n...
2	http://www.lvmama.com/trip/show/883678	奔跑的小东东	2021-02-26	2月出游	72张照片	海南岛|这里有数不尽的仲夏夜之梦	章节目录\n\n...	前言\n\n\n...	前言\n\n\n...
3	http://www.lvmama.com/trip/show/881878	游走旅行的摄影人	2020-12-16	11月出游	165张照片	海南五天四晚小众旅游地，躲开人挤人遇上碧海蓝天	章节目录\n\n...	前言\n\n\n...	前言\n\n\n...
4	http://www.lvmama.com/trip/show/880420	紫荆紫荆	2020-10-26	12天	69张照片	【海南宝藏攻略】吃黎家美食、穿越钻石级热带雨林	章节目录\n\n...	前言\n\n\n...	前言\n\n\n...

In [8]:

```
data2['作者去了这些地方'].dropna(how='any', inplace=True)
data2['作者去了这些地方']
```

Out[8]:

```
0       三亚-海棠湾-椰梦长廊-亚龙湾-后海村-亚特兰蒂斯水世界-南山文化旅游区-三亚国际免税城-三...
1                                  第一市场-三亚-亚龙湾-海棠湾-热带雨林-热带雨林
2                             三亚-落笔洞-水稻国家公园-海棠湾-红树林-红树林度假-皇冠假日
3       三亚-三亚湾-海棠湾-亚特兰蒂斯水世界-清水湾-清水湾-椰梦长廊-一线海景-大东海-第一市场...
5         三亚-蜈支洲岛-情人岛-岛海钓会所-崖州湾-海棠湾-亚龙湾-三亚悦榕庄-情人湾-三亚国际免税城
                                    ...
171                                          陵水-南湾猴岛
172                                           三亚-海棠湾
177                                          三亚-蜈支洲岛
187                                        三亚-亚特兰蒂斯水世界
189                               琼海-博鳌-万泉河-乐会古城-蔡家宅-博鳌禅寺-莲花墩
Name: 作者去了这些地方, Length: 98, dtype: object
```

In [9]:

```
def qfsj(x):
    list2 = []
    x = str(x)
    x = x.split('-')
    for i in range(len(x)-1):
        lx = str(x[i]) + "-" + str(x[i+1])
        list2.append(lx)
    return list2
```

In [13]:

```
# fp = open('./data/路线.txt','r',encoding='utf8')
# for line in fp:
#     fq = open('./data/路线1.txt','a',encoding='utf8')#这里用追加模式
#     fq.write(line)
# fp.close()
# fq.close()
```

In [14]:

```
# with open('./data/路线1.txt','w',encoding='utf-8')as f:
#     for d in data2['作者去了这些地方']:
#         f.write(str(d)+'\n')
```

这里上面的内容都是为了获取线路的内容，并且将线路合并在一起，然后再去计算它们的权重

In [15]:

```
import codecs
import numpy as np
import pandas as pd
import networkx as nx
import matplotlib.pyplot as plt
import csv
from scipy.sparse import coo_matrix
```

In [16]:

```
word = [] #记录关键词
f = open("./data/路线1.txt", encoding='utf-8')
line = f.readline()
while line:
    #print line
    line = line.replace("\n", "") #过滤换行
    line = line.strip('\n')
    for n in line.split('-'):
        #print n
        if n not in word:
            word.append(n)
    line = f.readline()
f.close()
print(len(word)) #关键词总数
```

187

In [17]:

```
word_vector = coo_matrix((len(word),len(word)), dtype=np.int8).toarray()
print(word_vector.shape)
```

(187, 187)

In [18]:

```
f = open("./data/路线1.txt", encoding='utf-8')
line = f.readline()
while line:
    line = line.replace("\n", "") #过滤换行
    line = line.strip('\n') #过滤换行
    nums = line.split('-')
    #循环遍历关键词所在位置 设置word_vector计数
    i = 0
    j = 0
    while i<len(nums):          #ABCD共现 AB AC AD BC BD CD加1
        j = i + 1
        w1 = nums[i]            #第一个单词
        while j<len(nums):
            w2 = nums[j]        #第二个单词
            #从word数组中找到单词对应的下标
            k = 0
            n1 = 0
            while k<len(word):
                if w1==word[k]:
                    n1 = k
                    break
                k = k +1
            #寻找第二个关键字位置
            k = 0
            n2 = 0
            while k<len(word):
                if w2==word[k]:
                    n2 = k
                    break
                k = k +1
            #重点: 词频矩阵赋值 只计算上三角
            if n1<=n2:
                word_vector[n1][n2] = word_vector[n1][n2] + 1
            else:
                word_vector[n2][n1] = word_vector[n2][n1] + 1
            #print n1, n2, w1, w2
            j = j + 1
        i = i + 1
    #读取新内容
    line = f.readline()
f.close()
```

In [19]:

```
res = open("./data/word_word_weight.txt", "a+", encoding='utf-8')
i = 0
while i<len(word):
    w1 = word[i]
    j = 0
    while j<len(word):
        w2 = word[j]
        #判断两个词是否共现 共现&词频不为0的写入文件
        if word_vector[i][j]>0:
            #print w1 +" " + w2 + " "+ str(int(word_vector[i][j]))
            res.write(w1 +" " + w2 + " "+ str(int(word_vector[i][j]))  +  "\n")
        j = j + 1
    i = i + 1
res.close()
```

In [20]:

```
c = open("./data/word-word-weight.csv","w", encoding='utf-8', newline='')    #解决空行
#c.write(codecs.BOM_UTF8)                                    #防止乱码
writer = csv.writer(c)                                       #写入对象
writer.writerow(['Word1', 'Word2', 'Weight'])

i = 0
while i<len(word):
    w1 = word[i]
    j = 0
    while j<len(word):
        w2 = word[j]
        #判断两个词是否共现 共现词频不为0的写入文件
        if word_vector[i][j]>0:
            #写入文件
            templist = []
            templist.append(w1)
            templist.append(w2)
            templist.append(str(int(word_vector[i][j])))
            #print templist
            writer.writerow(templist)
        j = j + 1
    i = i + 1
c.close()
```

根据文章上面的计算该地方到另一个地方的权重，然后通过得到它们的权重去画出它们的空间流动图

In [21]:

```
df5 = pd.read_csv('./data/word-word-weight.csv')
df5.sort_values("Weight",inplace=False)
df5.head()
```

Out[21]:

	Word1	Word2	Weight
0	三亚	海棠湾	64
1	三亚	椰梦长廊	34
2	三亚	亚龙湾	57
3	三亚	后海村	4
4	三亚	亚特兰蒂斯水世界	7

In [22]:

```
df5 = df5[df5['Weight'] >=5]
df5 = df5.replace("三亚国际免税城","三亚免税店").replace("红树林度假","红树林")
```

In [23]:

```
word1 = df5.groupby(['Word1']).sum()
word1['Word'] = word1.index
```

In [24]:

```
word2 = df5.groupby(['Word2']).sum()
word2['Word'] = word2.index
```

统计每一个地方权重的总和

In [25]:

```
word_results = pd.concat([word1,word2])
word_results = word_results.groupby(['Word']).sum()
word_results
```

Out[25]:

Word	Weight
三亚	514
三亚免税店	98
三亚河	46
三亚湾	427
亚特兰蒂斯水世界	12
亚龙湾	428
亚龙湾热带天堂森林公园	42
凤凰岛	56
半山半岛帆船港	71
南天一柱	42
南山寺	77
南山文化旅游区	15
后海	73
喜来登	99
大东海	336
大小洞天	91
天涯海角	221
小东海	34
希尔顿	10
情人岛	22
情人桥	143
椰梦长廊	281
海月广场	10
海棠湾	456
清水湾	5
热带雨林	16
第一市场	161
红树林	55
蜈支洲岛	199

Word	Weight
西岛	102
香水湾	10

In [26]:

```
categories = []
for d in word_results.index:
    cate_dict = {
        'name':d
    }
    categories.append(cate_dict)
```

In [66]:

```
nodes_list = []
count = 0
for j in range(len(word_results.index.to_list())):
    nodes = {
            "id": '{}'.format(count),
            'name':word_results.index.to_list()[j],
            "symbolSize":float(int(word_results.Weight.to_list()[j]) / 10),
            "value":float(int(word_results.Weight.to_list()[j]) / 10),
            "category":count
    }
    count += 1
    nodes_list.append(nodes)
```

In [67]:

```
dic = {}
count = 0
for w in word_results.index.to_list():
    dic[w] = count
    count += 1

def thsz(x):
    for key,values in dic.items():
        if x == key:
            return values
```

In [68]:

```
links_list = []
for j in range(len(df5.Word1.to_list())):
    links = {
            'source':'{}'.format(thsz(df5.Word1.to_list()[j])),
            'target':'{}'.format(thsz(df5.Word2.to_list()[j])),
        }
    links_list.append(links)
```

In [69]:

```
from pyecharts import options as opts
from pyecharts.charts import Graph
import json

c = (
    Graph(init_opts=opts.InitOpts(width="1000px", height="600px"))
    .add(
        "",
        nodes=nodes_list,
        links=links_list,
        categories=categories,
        layout="circular",
        is_rotate_label=True,
        linestyle_opts=opts.LineStyleOpts(color="source", curve=0.3),
        label_opts=opts.LabelOpts(position="right"),
    )
    .set_global_opts(
        title_opts=opts.TitleOpts(title="旅游流的空间转移分析"),
        legend_opts=opts.LegendOpts(is_show=False),
    )
)
```

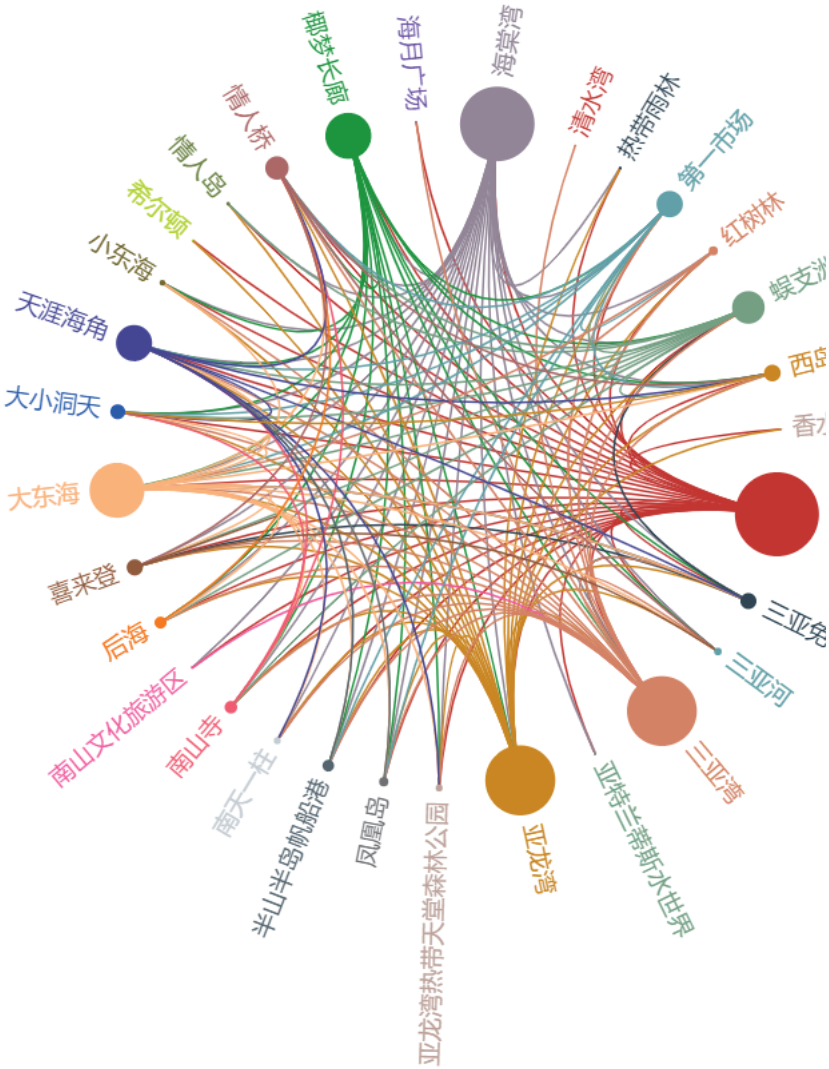
然后画出这个地点到另一个地点的空间转移分析图

In [70]:

```
c.render_notebook()
```

Out[70]:

旅游流的空间转移分析



In [245]:

```
c = (
    Graph(init_opts=opts.InitOpts(width="1000px", height="600px"))
    .add(
        "",
        nodes=nodes_list,
        links=links_list,
        categories=categories,
        layout="circular",
        is_rotate_label=True,
        linestyle_opts=opts.LineStyleOpts(color="source", curve=0.3),
        label_opts=opts.LabelOpts(position="right"),
    )
    .set_global_opts(
        title_opts=opts.TitleOpts(title="旅游流的空间转移分析"),
        legend_opts=opts.LegendOpts(is_show=False),
    )
    .render("旅游流的空间转移分析.html")
)
```

这一块是计算它们的出游动机，根据一些关键词的存在，判断出它们的出游动机属于哪一列

In [54]:

```
def cydj(x):
    x = str(x)
    if '放松' in x or '度假' in x or '休闲' in x or '海滩' in x or '观光' in x or '假日' in x:
        return '徒步、登山、骑行'
    elif '开会' in x or '会展' in x or '会议' in x or '商务' in x:
        return '商务会议'
    elif '宗教' in x or '教堂' in x or '佛' in x or '观音' in x:
        return '宗教'
    elif '身体' in x or '健康' in x or '心智' in x or '保养' in x or '修养' in x or '养生' in x:
        return '康养旅游'
    elif '文化' in x or '人文' in x or '苗族' in x or '少数民族' in x or '黎族' in x:
        return '文化旅游'
    elif '登山' in x or '徒步' in x or '探险' in x or '自行骑行' in x or '露营' in x:
        return '徒步、登山、骑行'
    elif '乡村' in x or '农村' in x or '村' in x or '农场' in x:
        return '乡村旅游'
    elif '体育' in x or '航空' in x or '科技' in x or '发射' in x or '体育馆' in x or '火箭' in
        return '体育、科技'
    else:
        return '其他'
```

In [55]:

```
df = pd.read_excel('./data/clean_data.xlsx')
```

In [56]:

```
df['出游动机'] = df['内容'].apply(cydj)
```

In [57]:

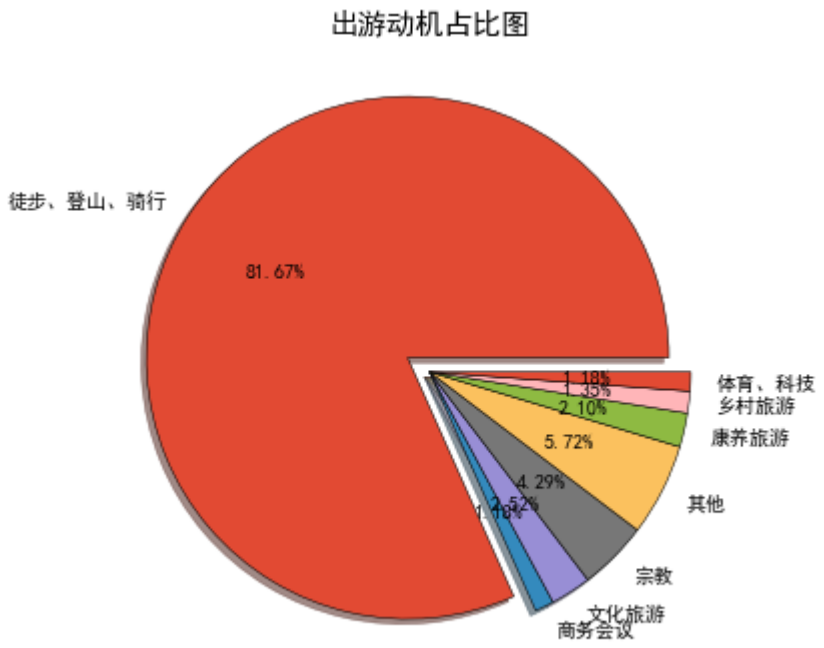
```
counts = {}
for word in df['出游动机']:
    counts[word] = counts.get(word,0) + 1
```

In [60]:

```
list_key = []
list_values = []
for key,values in counts.items():
    list_key.append(key)
    list_values.append(values)
```

In [61]:

```
plt.rcParams['font.sans-serif'] = ['SimHei']
plt.style.use('ggplot')
plt.figure(figsize=(12,6))
explo = [0.1,0,0,0,0,0,0,0]
plt.pie(list_values,labels=list_key,explode=explo,shadow=True,startangle=0,autopct='%1.2f%%',wedgep
plt.title('出游动机占比图')
plt.savefig('出游动机占比图.jpg')
plt.show()
```



后根据上面的统计，最后画出出游动机的饼图

下面这一块是计算它们的逗留天数

In [252]:

```
def tsjs(x):
    x = str(x)
    x = x.replace('天','').strip(" ")
    if x == np.nan:
        return 0
    else:
        return x
data1['逗留天数'] = data1['逗留天数'].apply(tsjs)
data2['天数'] = data2['天数'].apply(tsjs)
data3['出行天数'] = data3['出行天数'].apply(tsjs)
```

In [253]:

```
data = pd.concat([data1['逗留天数'],data2['天数'],data3['出行天数']])
counts = {}
for number in data.values:
    counts[number] = counts.get(number,0) + 1
```

In [254]:

```
counts.pop('nan')
```

Out[254]:

489

In [255]:

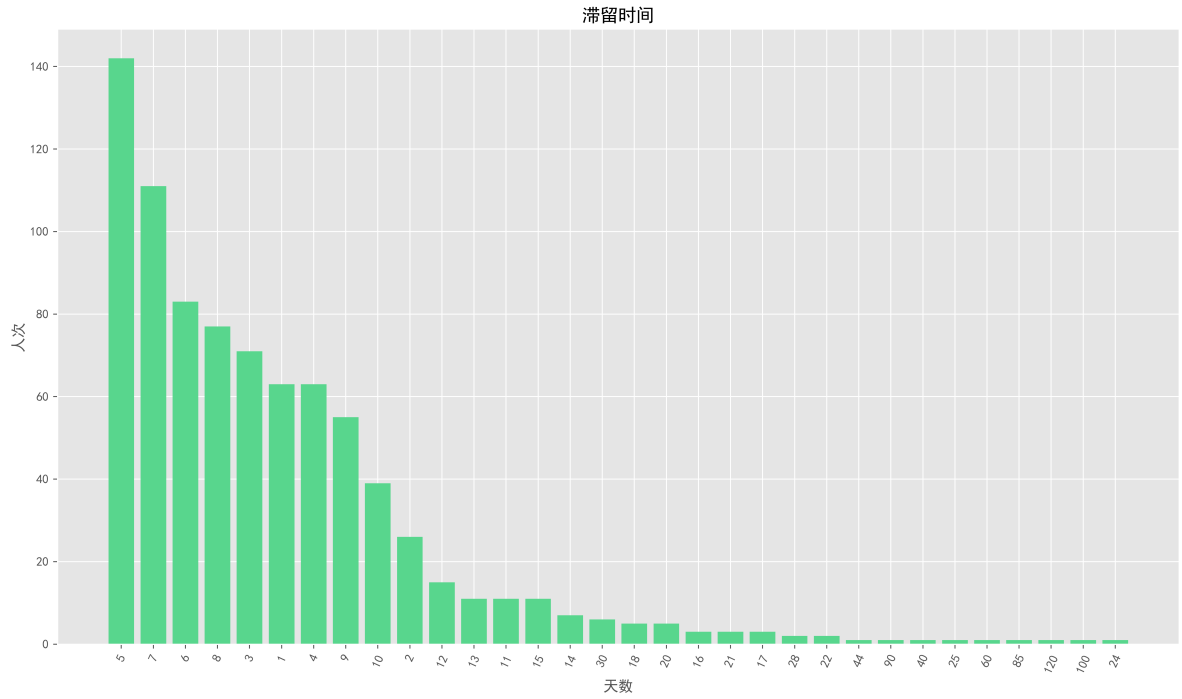
```
ls = list(counts.items())
ls.sort(key=lambda x:x[1],reverse=True)
```

In [256]:

```
list_key = []
list_values = []
for key,values in ls:
    list_key.append(key)
    list_values.append(values)
```

In [257]:

```
plt.style.use('ggplot')
plt.figure(figsize=(16,9),dpi=500)
plt.bar(list_key,list_values,color='#58D68D')
plt.rcParams['font.sans-serif'] = ['SimHei']
plt.title("滞留时间")
plt.xlabel("天数")
plt.ylabel("人次")
plt.xticks(rotation=65)
plt.savefig('滞留时间.jpg')
plt.show()
```



根据每个人逗留天数的统计，来画出柱状图，得知一般逗留的天数都是在10天以内

In [27]:

```
#数据处理库
import numpy as np
import pandas as pd
import glob
import re
import jieba

#可视化库
import stylecloud
import matplotlib.pyplot as plt
import seaborn as sns
%matplotlib inline
from pyecharts.charts import *
from pyecharts import options as opts
from pyecharts.globals import ThemeType
from IPython.display import Image
df2 = pd.read_excel('./data/2.xlsx')
data_pair_1 = [(i, int(j)) for i, j in zip(df2['word'], df2['count'])]
import pyecharts.options as opts
from pyecharts.charts import WordCloud
from pyecharts.globals import ThemeType
w = (
    WordCloud(init_opts=opts.InitOpts(theme=ThemeType.LIGHT,width="600px",height="400px"))
    .add(series_name="热词-云图", data_pair=data_pair_1[0:100],word_size_range=[15, 66])
    .set_global_opts(
        title_opts=opts.TitleOpts(
            title="热词-云图", title_textstyle_opts=opts.TextStyleOpts(font_size=23)
        ),
        tooltip_opts=opts.TooltipOpts(is_show=True),
    )
)
w.render_notebook()
# # 定义分词函数
# def get_cut_words(content_series):
#     # 读入停用词表
#     stop_words = []

#     with open("./data/stopwords_cn.txt", 'r', encoding='utf-8') as f:
#         lines = f.readlines()
#         for line in lines:
#             stop_words.append(line.strip())

#     # 分词
#     word_num = jieba.lcut(content_series.str.cat(sep='。'), cut_all=False)

#     # 条件筛选
#     word_num_selected = [i for i in word_num if i not in stop_words and len(i)>=2]
#     return word_num_selected

# # 绘制词云图
# text1 = get_cut_words(content_series=df2['word'])
# stylecloud.gen_stylecloud(text=' '.join(text1), max_words=100,
#                           collocations=False,
#                           font_path='simhei.ttf',
#                           icon_name='fas fa-heart',
#                           size=500,
#                           #palette='matplotlib.Inferno_9',
#                           output_name='热词-云图.png')
# Image(filename='热词-云图.png')
```

Out[27]:

热词-云图

上面这一块是则是画出高频词的词云图

In [262]:

```
# 定义分词函数
def get_cut_words(content_series):
    # 读入停用词表
    stop_words = []

    with open("./data/stopwords_cn.txt", 'r', encoding='utf-8') as f:
        lines = f.readlines()
        for line in lines:
            stop_words.append(line.strip())

    # 分词
    word_num = jieba.lcut(content_series.str.cat(sep='。'), cut_all=False)

    # 条件筛选
    word_num_selected = [i for i in word_num if i not in stop_words and len(i)>=2]
    for word in word_num_selected:
        for w in word:
            list_word.append(w)
    return list_word
```

In [345]:

```
ditc = {}
list_word = []
list_count = []
for t in text1:
    ditc[t] = ditc.get(t, 0) + 1
ls = list(ditc.items())
ls.sort(key=lambda x: x[1], reverse=True)
for i in range(len(ls)):
    word, count = ls[i]
    list_word.append(word)
    list_count.append(count)

df1 = pd.DataFrame()

df1['word'] = list_word
df1['count'] = list_count
df1.to_csv('./data/高频词.csv',encoding='gbk')
```

上面这一块则是统计所以高频词的词频次数，最后用CSV文件把数据保存下来

这里是把上面高频词的前100个进行共现矩阵的计算去统计词与词之间的联系

In [28]:

```
""" 第三步:共现矩阵计算 共现词频 文件 """
list_word = df2['word']
words = codecs.open("./data/word_node.txt", "a+", "utf-8")
i = 0
while i<len(list_word[0:50]):  #len(name)
    student1 = list_word[i]
    j = i + 1
    while j<len(list_word[0:50]):
        student2 = list_word[j]
        #判断学生是否共现 共现词频不为0则加入
        if word_vector[i][j]>0:
            words.write(student1 + " " + student2 + " "
                + str(word_vector[i][j]) + "\r\n")
        j = j + 1
    i = i + 1
words.close()
```

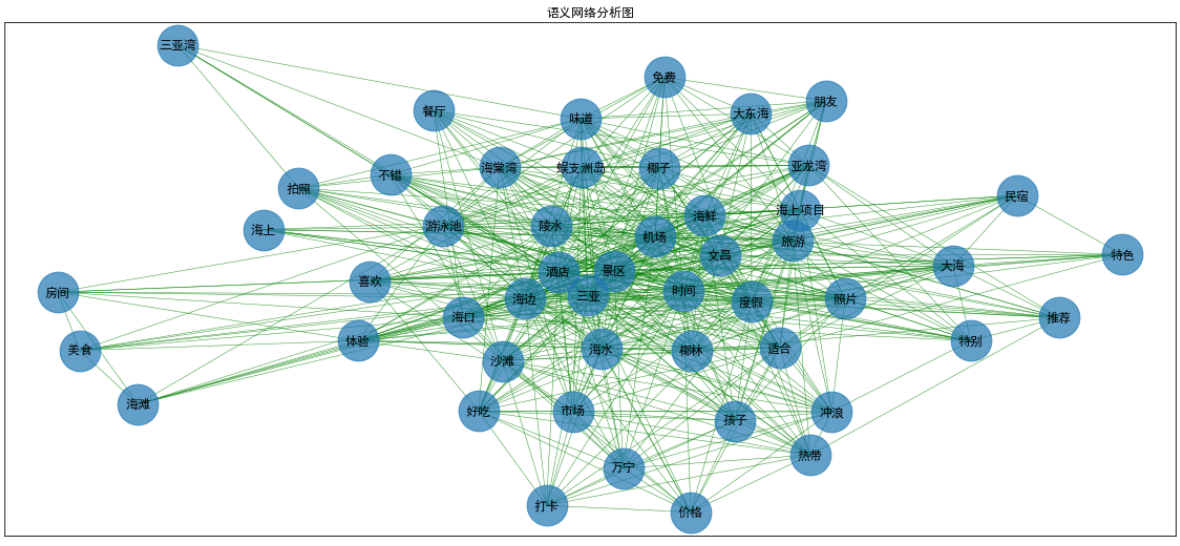
In [29]:

```
""" 第四步:图形生成 """
a = []
f = codecs.open('./data/word_node.txt','r','utf-8')
line = f.readline()
i = 0
A = []
B = []
while line!="":
    a.append(line.split())    #保存文件是以空格分离的
    A.append(a[i][0])
    B.append(a[i][1])
    i = i + 1
    line = f.readline()
elem_dic = tuple(zip(A,B))
f.close()
```

In [30]:

```
import matplotlib
colors = ["red","green","blue","yellow"]
G = nx.Graph()
G.add_edges_from(list(elem_dic))
matplotlib.rcParams['font.sans-serif'] = ['SimHei']
matplotlib.rcParams['font.family']='sans-serif'
plt.figure(figsize=(20, 9))
pos=nx.spring_layout(G,iterations=100)
nx.draw_networkx_nodes(G, pos, alpha=0.7,node_size=1500)
nx.draw_networkx_edges(G,pos,width=0.5,alpha=0.8,edge_color='g')
nx.draw_networkx_labels(G, pos, font_family='sans-serif', alpha=1)
plt.title("语义网络分析图")
plt.savefig('语义网络分析图.jpg')
plt.show()
```

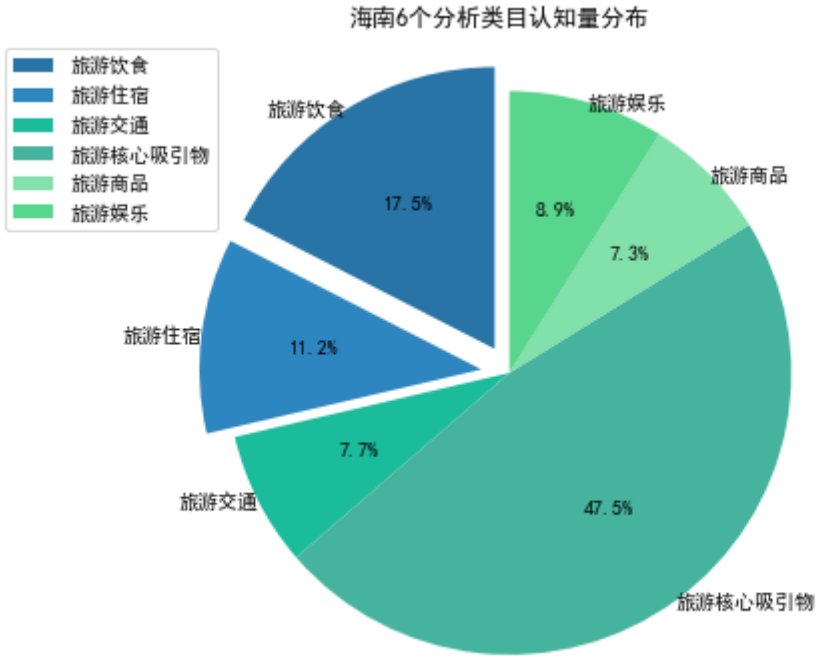
```
D:\rgzn\lib\site-packages\networkx\drawing\nx_pylab.py:579: MatplotlibDeprecationWar
ning:
The iterable function was deprecated in Matplotlib 3.1 and will be removed in 3.3. U
se np.iterable instead.
  if not cb.iterable(width):
```



最后就是通过上面的关联来生成语义网络分析图

In [51]:

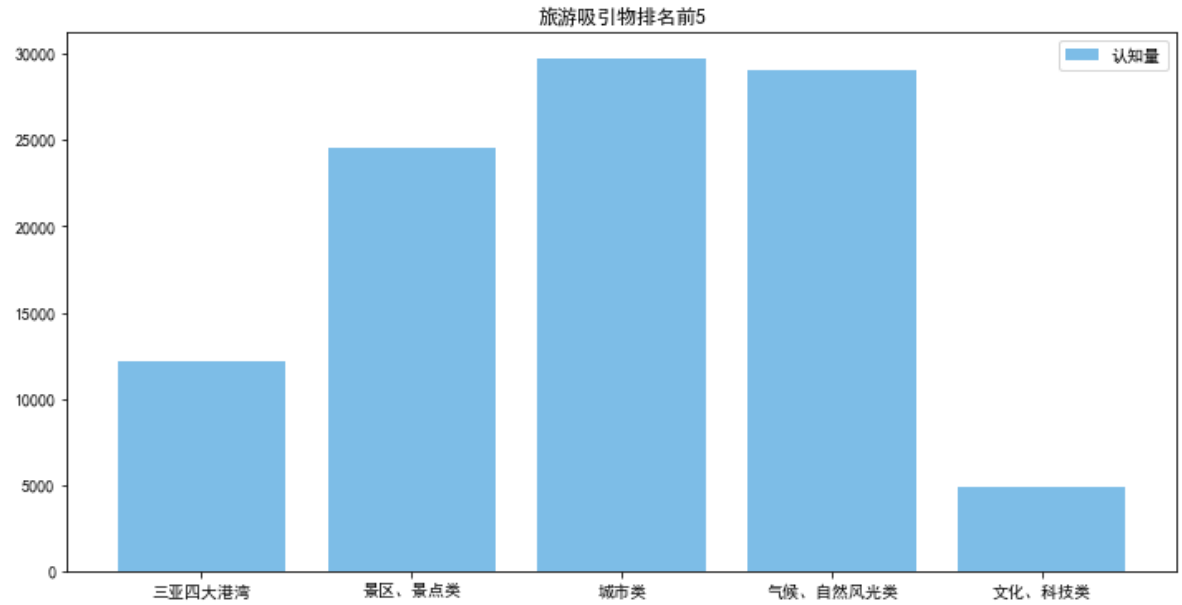
```
x_data = ['旅游吸引物','旅游交通','旅游住宿','旅游娱乐','旅游饮食','旅游商品']
y_data = [54085,2741,16963,10659,12187,8832]
df3 = pd.read_excel('./data/1.xlsx').loc[:408,['word','count','labels']]
sum_labels_1 = 0
sum_labels_2 = 0
sum_labels_3 = 0
sum_labels_4 = 0
sum_labels_5 = 0
sum_labels_6 = 0
for j,k,l in zip(df3.word.to_list(),df3['count'].to_list(),df3.labels.to_list()):
    if int(l) == 1:
        sum_labels_1 += k
    if int(l) == 2:
        sum_labels_2 += k
    if int(l) == 3:
        sum_labels_3 += k
    if int(l) == 4:
        sum_labels_4 += k
    if int(l) == 5:
        sum_labels_5 += k
    if int(l) == 6:
        sum_labels_6 += k
x_data = ['旅游饮食','旅游住宿','旅游交通','旅游核心吸引物','旅游商品','旅游娱乐']
y_data = [sum_labels_1,sum_labels_2,sum_labels_3,sum_labels_4,sum_labels_5,sum_labels_6]
matplotlib.rcParams['font.sans-serif'] = ['SimHei']
matplotlib.rcParams['font.family']='sans-serif'

plt.figure(figsize=(9,6)) #调节图形大小
labels = x_data #定义标签
sizes = y_data #每块值
colors = ['#2874A6','#2E86C1','#1ABC9C','#45B39D','#82E0AA','#58D68D'] #每块颜色定义
explode = (0.1,0.1,0,0,0,0) #将某一块分割出来，值越大分割出的间隙越大
patches,text1,text2 = plt.pie(sizes,
                      explode=explode,
                      labels=labels,
                      colors=colors,
                      labeldistance = 1,#图例距圆心半径倍距离
                      autopct = '%3.1f%%', #数值保留固定小数位
                      shadow = False, #无阴影设置
                      startangle =90, #逆时针起始角度设置
                      pctdistance = 0.6) #数值距圆心半径倍数距离
#patches饼图的返回值，texts1饼图外label的文本，texts2饼图内部文本
# x，y轴刻度设置一致，保证饼图为圆形
plt.axis('equal')
plt.title('海南6个分析类目认知量分布')
plt.legend()
plt.savefig('海南6个分析类目认知量分布.jpg')
plt.show()
```



这个饼图的生成是基于高频词，然后把里面的词进行归类，最后统计出来的饼图

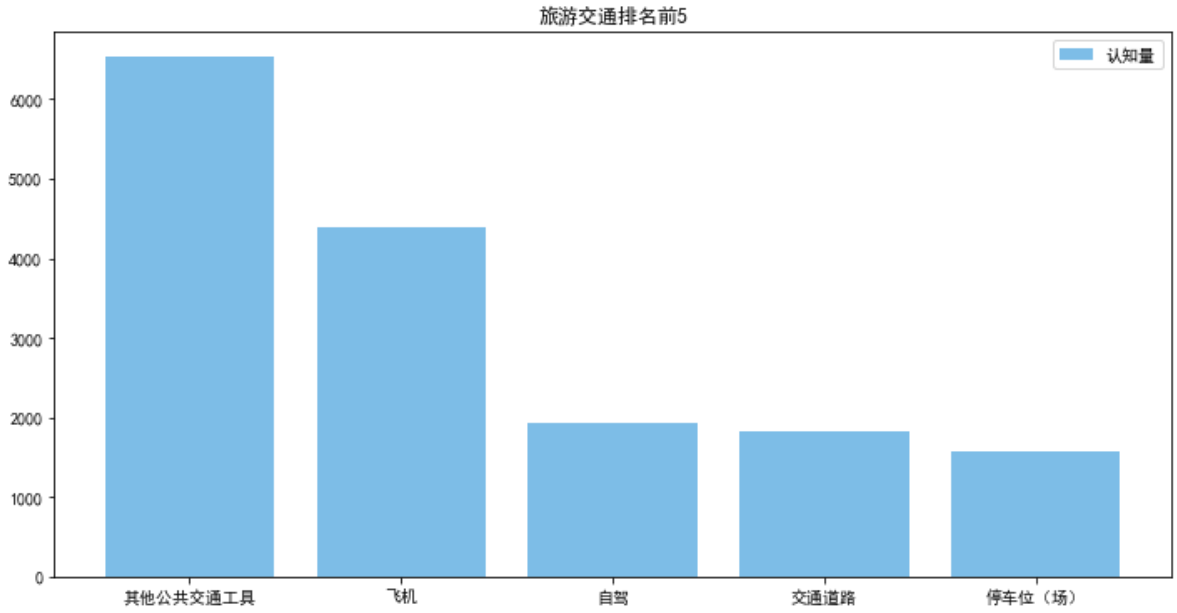
In [59]:

```
import matplotlib.ticker as mtick
#旅游吸引物
x_data1 = ['三亚四大港湾','景区、景点类','城市类','气候、自然风光类','文化、科技类']
y_data1 = [12163,24570,29758,29048,4945]
# def bfb(y_data):
#     y_d = []
#     for y in y_data:
#         a = y / sum(y_data)
#         a = "%0.2lf" %a
#         y_d.append(float(a))
#     return y_d
# y_data = bfb(y_data1)
# fmt='%.2f%%'
# yticks = mtick.FormatStrFormatter(fmt)  #设置百分比形式的坐标轴
# fig = plt.figure()
# ax1 = fig.add_subplot(111)
# ax1.plot(x_data1, y_data,'or-',label='认知强度');
# ax1.yaxis.set_major_formatter(yticks)

# ax1.set_ylim()
# ax1.set_ylabel('认知强度百分比(%)')

plt.figure(figsize=(12,6))
plt.bar(x_data1,y_data1,color='#5DADE2',label='认知量',alpha=0.8)
plt.title('旅游吸引物排名前5')
plt.legend()
plt.savefig('旅游吸引物.png')
plt.show()
```



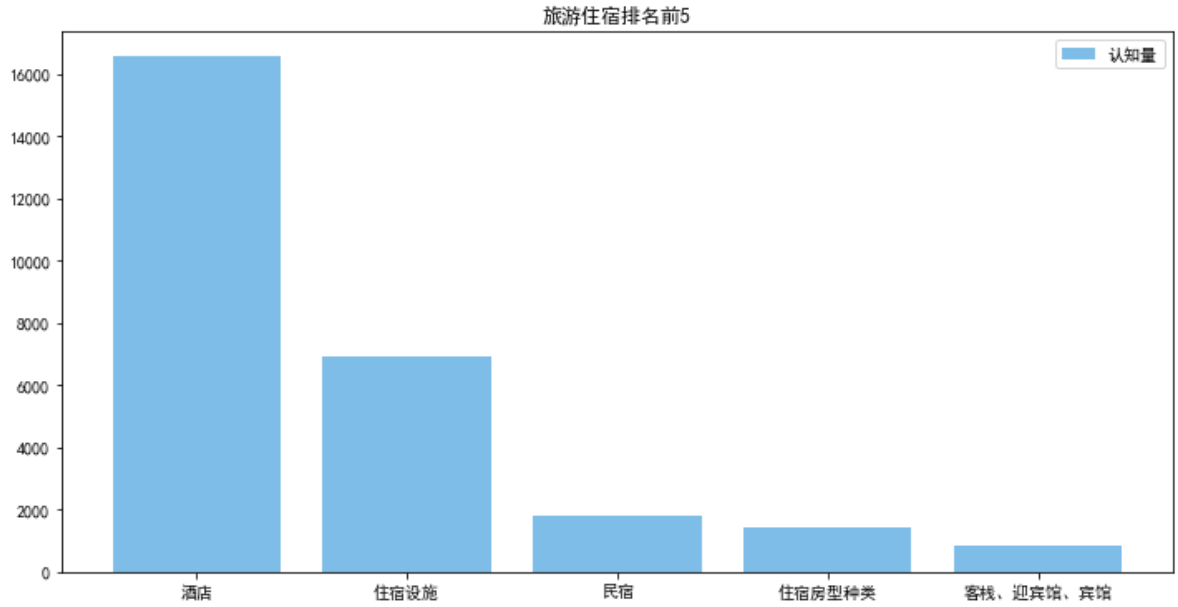
In [60]:

```
#旅游交通
x_data2 = ['其他公共交通工具','飞机','自驾','交通道路','停车位（场）']
y_data2 = [6528,4388,1936,1824,1574]

# def bfb(y_data):
#     y_d = []
#     for y in y_data:
#         a = y / sum(y_data)
#         a = "%0.2lf" %a
#         y_d.append(float(a))
#     return y_d
# y_data = bfb(y_data2)
# fmt='%.2f%%'
# yticks = mtick.FormatStrFormatter(fmt)  #设置百分比形式的坐标轴
# fig = plt.figure()
# ax1 = fig.add_subplot(111)
# ax1.plot(x_data2, y_data,'or-',label='认知强度');
# ax1.yaxis.set_major_formatter(yticks)

# ax1.set_ylim()
# ax1.set_ylabel('认知强度百分比(%)')

# ax2 = ax1.twinx() # this is the important function

plt.figure(figsize=(12,6))
plt.bar(x_data2,y_data2,color='#5DADE2',label='认知量',alpha=0.8)
plt.title('旅游交通排名前5')
plt.legend()
plt.savefig('旅游交通.png')
plt.show()
```



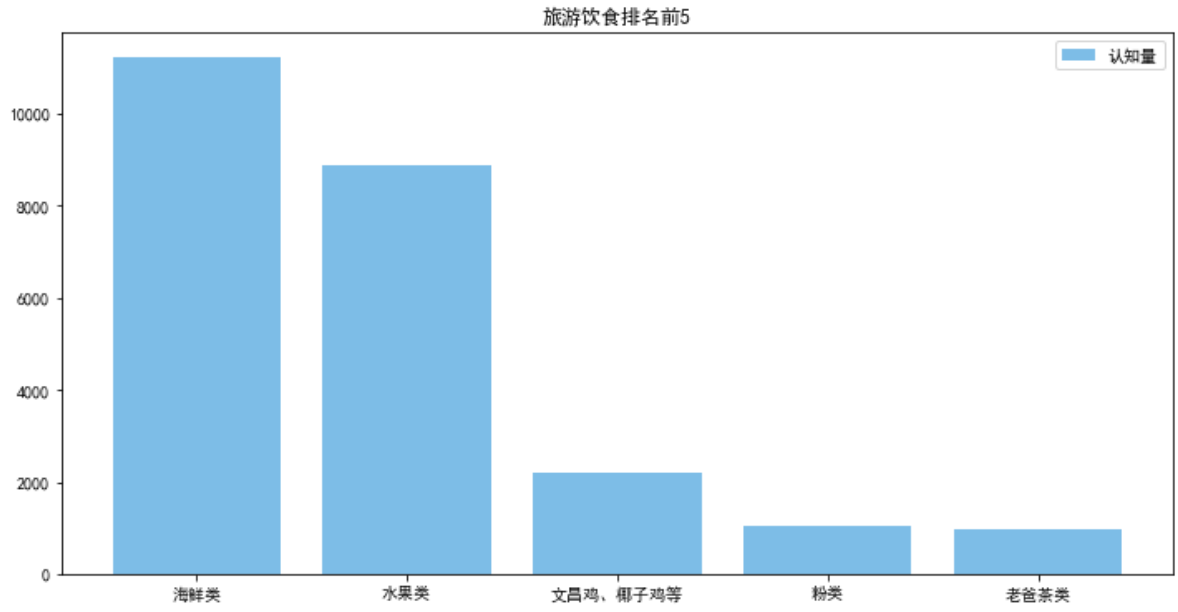
In [62]:

```
#旅游住宿
x_data3 = ['酒店','住宿设施','民宿','住宿房型种类','客栈、迎宾馆、宾馆']
y_data3 = [16549,6945,1785,1457,830]

# def bfb(y_data):
#     y_d = []
#     for y in y_data:
#         a = y / sum(y_data)
#         a = "%0.2lf" %a
#         y_d.append(float(a))
#     return y_d
# y_data = bfb(y_data3)
# fmt='%.2f%%'
# yticks = mtick.FormatStrFormatter(fmt)  #设置百分比形式的坐标轴
# fig = plt.figure()
# ax1 = fig.add_subplot(111)
# ax1.plot(x_data3, y_data,'or-',label='认知强度');
# ax1.yaxis.set_major_formatter(yticks)

# ax1.set_ylim()
# ax1.set_ylabel('认知强度百分比(%)')

# ax2 = ax1.twinx() # this is the important function
# plt.bar(x_data3,y_data3,color='#5DADE2',label='认知量',alpha=0.4)
# ax1.legend(loc=1)
# ax2.legend(loc=9)
# ax2.set_ylabel('认知量')

plt.figure(figsize=(12,6))
plt.bar(x_data3,y_data3,color='#5DADE2',label='认知量',alpha=0.8)
plt.title('旅游住宿排名前5')
plt.legend()
plt.savefig('旅游住宿.png')
plt.show()
```



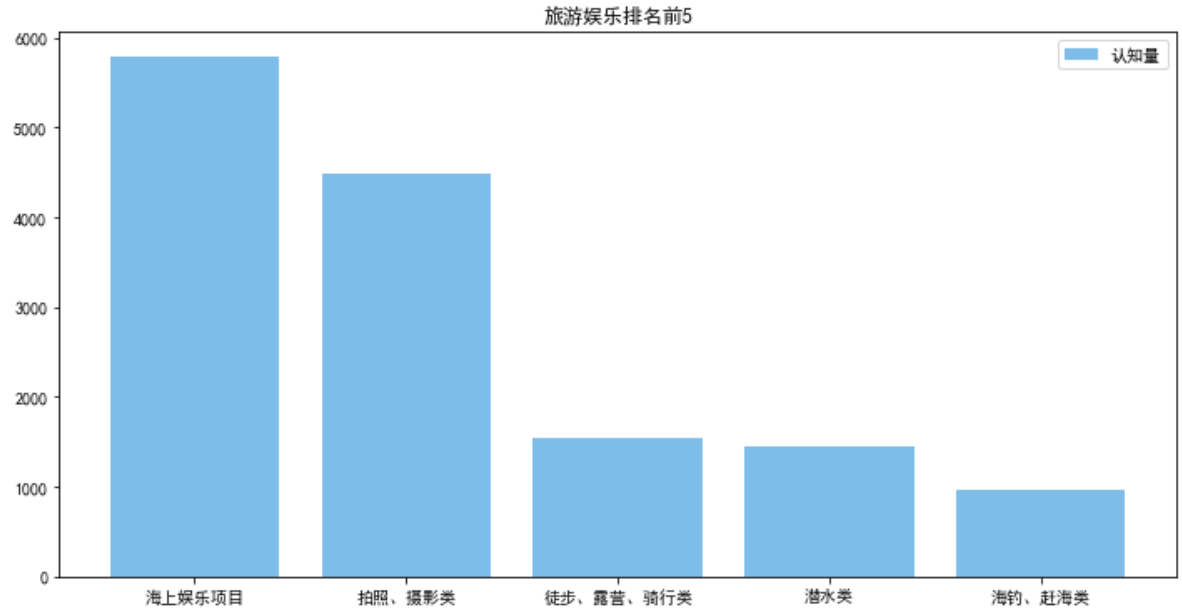
In [63]:

```
#旅游饮食
x_data4 = ['海鲜类','水果类','文昌鸡、椰子鸡等','粉类','老爸茶类']
y_data4 = [11209,8863,2201,1060,974]

# def bfb(y_data):
#     y_d = []
#     for y in y_data:
#         a = y / sum(y_data)
#         a = "%0.2lf" %a
#         y_d.append(float(a))
#     return y_d
# y_data = bfb(y_data4)
# fmt='%.2f%%'
# yticks = mtick.FormatStrFormatter(fmt)  #设置百分比形式的坐标轴
# fig = plt.figure()
# ax1 = fig.add_subplot(111)
# ax1.plot(x_data4, y_data,'or-',label='认知强度');
# ax1.yaxis.set_major_formatter(yticks)

# ax1.set_ylim()
# ax1.set_ylabel('认知强度百分比(%)')

# ax2 = ax1.twinx() # this is the important function
# plt.bar(x_data4,y_data4,color='#5DADE2',label='认知量',alpha=0.4)
# ax1.legend(loc=1)
# ax2.legend(loc=9)
# ax2.set_ylabel('认知量')

plt.figure(figsize=(12,6))
plt.bar(x_data4,y_data4,color='#5DADE2',label='认知量',alpha=0.8)
plt.title('旅游饮食排名前5')
plt.legend()
plt.savefig('旅游饮食.png')
plt.show()
```



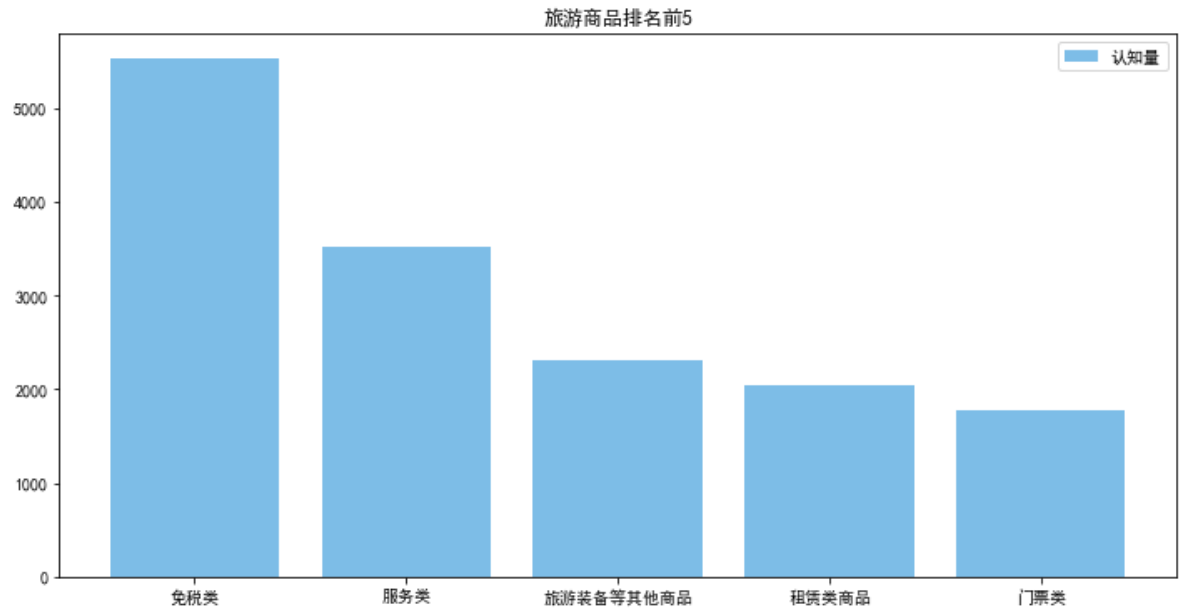
In [64]:

```
#旅游娱乐
x_data5 = ['海上娱乐项目','拍照、摄影类','徒步、露营、骑行类','潜水类','海钓、赶海类']
y_data5 = [5787,4478,1540,1446,974]

# def bfb(y_data):
#     y_d = []
#     for y in y_data:
#         a = y / sum(y_data)
#         a = "%0.2lf" %a
#         y_d.append(float(a))
#     return y_d
# y_data = bfb(y_data5)
# fmt='%.2f%%'
# yticks = mtick.FormatStrFormatter(fmt)  #设置百分比形式的坐标轴
# fig = plt.figure()
# ax1 = fig.add_subplot(111)
# ax1.plot(x_data5, y_data,'or-',label='认知强度');
# ax1.yaxis.set_major_formatter(yticks)

# ax1.set_ylim()
# ax1.set_ylabel('认知强度百分比(%)')

# ax2 = ax1.twinx() # this is the important function
# plt.bar(x_data5,y_data5,color='#5DADE2',label='认知量',alpha=0.4)
# ax1.legend(loc=1)
# ax2.legend(loc=9)
plt.figure(figsize=(12,6))
plt.bar(x_data5,y_data5,color='#5DADE2',label='认知量',alpha=0.8)
plt.title('旅游娱乐排名前5')
plt.legend()
plt.savefig('旅游娱乐.png')
plt.show()
```



In [65]:

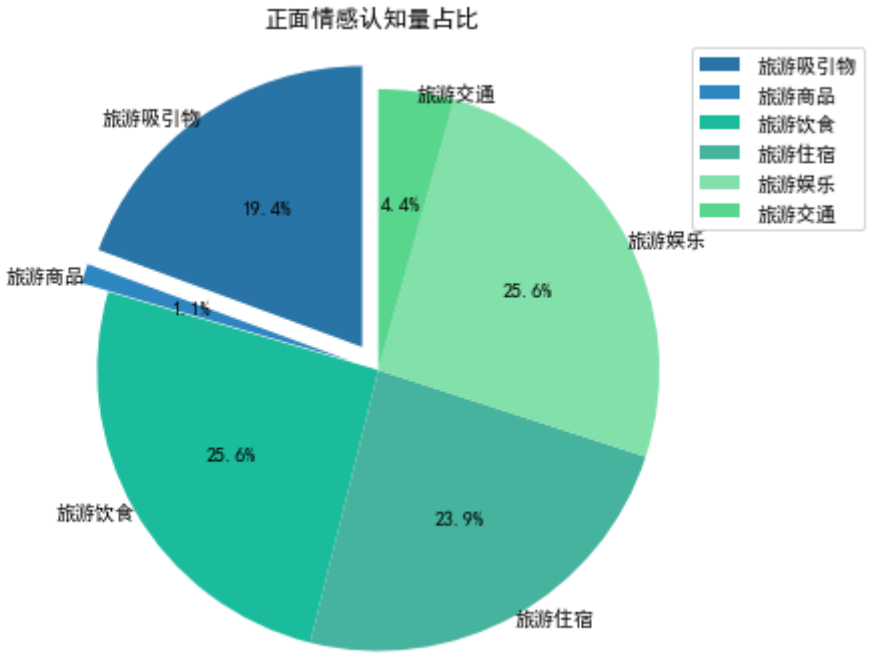
```
#旅游商品
x_data6 = ['免税类','服务类','旅游装备等其他商品','租赁类商品','门票类']
y_data6 = [5521,3515,2309,2049,1780]

# def bfb(y_data):
#     y_d = []
#     for y in y_data:
#         a = y / sum(y_data)
#         a = "%0.2lf" %a
#         y_d.append(float(a))
#     return y_d
# y_data = bfb(y_data6)
# fmt='%.2f%%'
# yticks = mtick.FormatStrFormatter(fmt)  #设置百分比形式的坐标轴
# fig = plt.figure()
# ax1 = fig.add_subplot(111)
# ax1.plot(x_data6, y_data,'or-',label='认知强度');
# ax1.yaxis.set_major_formatter(yticks)

# ax1.set_ylim()
# ax1.set_ylabel('认知强度百分比(%)')

# ax2 = ax1.twinx() # this is the important function
# plt.bar(x_data6,y_data6,color='#5DADE2',label='认知量',alpha=0.4)
# ax1.legend(loc=1)
# ax2.legend(loc=9)
# ax2.set_ylabel('认知量')
plt.figure(figsize=(12,6))
plt.bar(x_data6,y_data6,color='#5DADE2',label='认知量',alpha=0.8)
plt.title('旅游商品排名前5')
plt.legend()
plt.savefig('旅游商品.png')
plt.show()
```



上面的排名前五的图都是基于高频词来找寻最前的5个词，然后再去计算他们所占的百分比和总量最后生成的图形

In [44]:

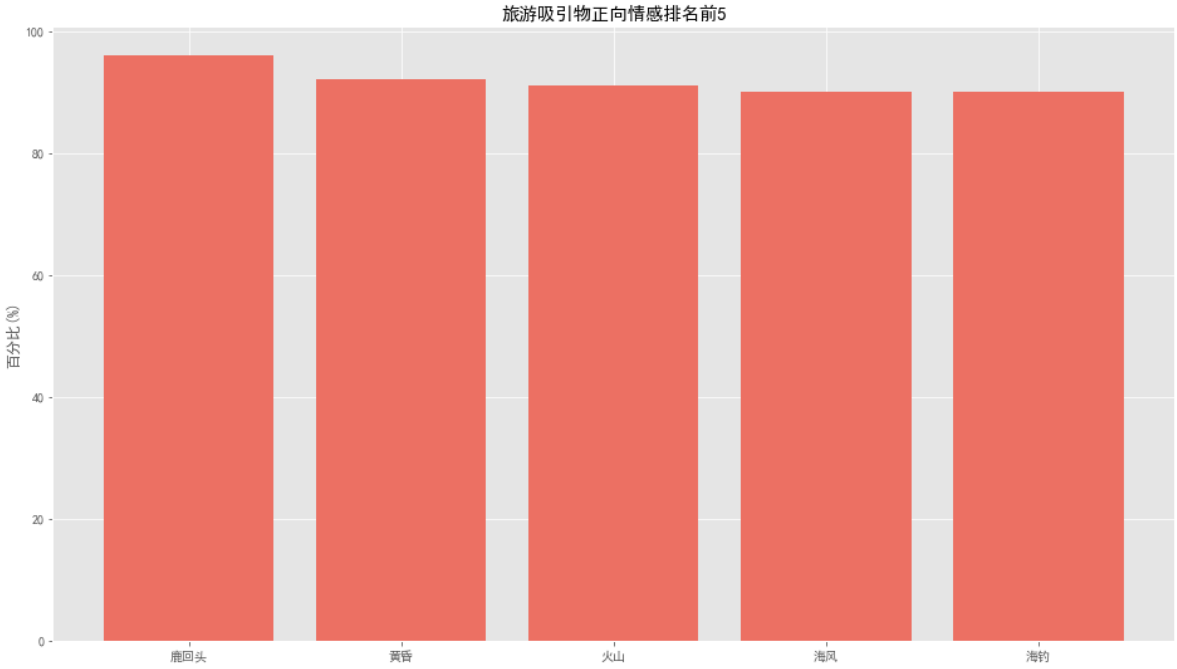
```
x_data = ['旅游吸引物','旅游商品','旅游饮食','旅游住宿','旅游娱乐','旅游交通']
y_data = [4595,269,6071,5666,6079,1033]

plt.figure(figsize=(9,6)) #调节图形大小
labels = x_data #定义标签
sizes = y_data #每块值
colors = ['#2874A6','#2E86C1','#1ABC9C','#45B39D','#82E0AA','#58D68D'] #每块颜色定义
explode = (0.1,0.1,0,0,0,0) #将某一块分割出来，值越大分割出的间隙越大
patches,text1,text2 = plt.pie(sizes,
                      explode=explode,
                      labels=labels,
                      colors=colors,
                      labeldistance = 1,#图例距圆心半径倍距离
                      autopct = '%3.1f%%', #数值保留固定小数位
                      shadow = False, #无阴影设置
                      startangle =90, #逆时针起始角度设置
                      pctdistance = 0.6) #数值距圆心半径倍数距离
#patches饼图的返回值，texts1饼图外label的文本，texts2饼图内部文本
# x，y轴刻度设置一致，保证饼图为圆形
plt.axis('equal')
plt.title('正面情感认知量占比')
plt.legend()
plt.savefig('正面情感认知量占比.jpg')
plt.show()
```



该饼图是基于正面情感TOP100高频词来对这些词进行归类，然后根据这些汇总的数据做成饼图

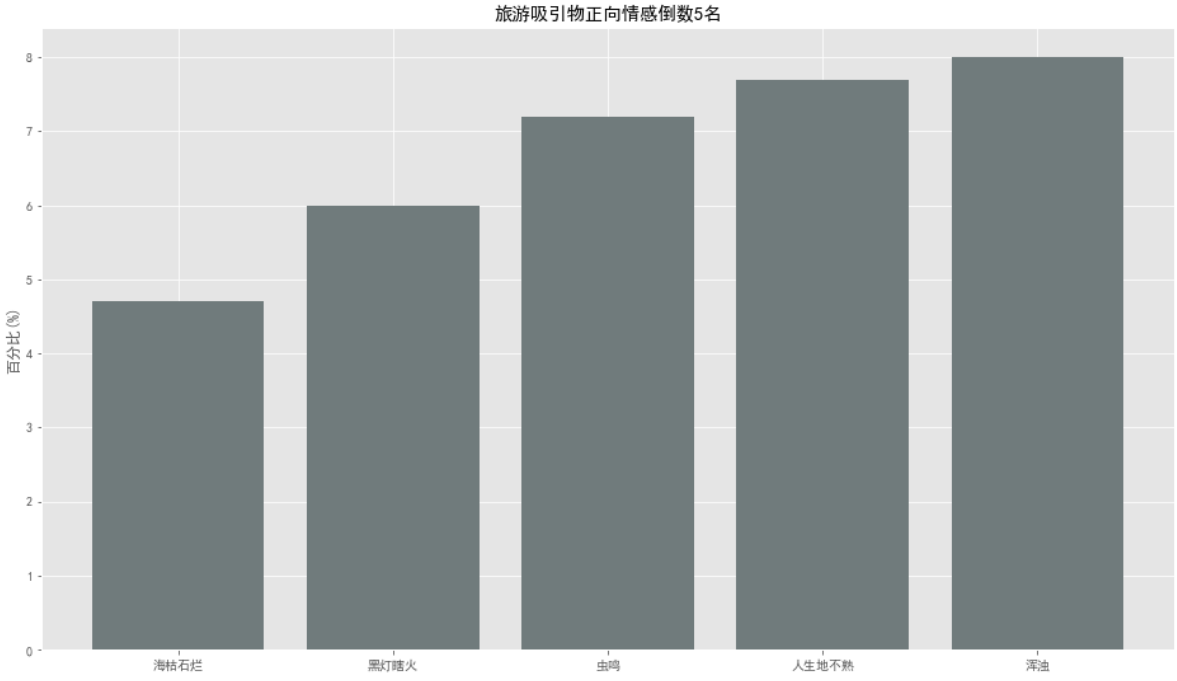
In [406]:

```
x_data10 = ['鹿回头','黄昏','火山','海风','海钓']
y_data10 = [96,92,91,90,90]
plt.style.use('ggplot')
plt.figure(figsize=(16,9))
plt.bar(x_data10,y_data10,color='#EC7063')
plt.rcParams['font.sans-serif'] = ['SimHei']
plt.title("旅游吸引物正向情感排名前5")
plt.ylabel("百分比(%)")
plt.savefig('旅游吸引物正向情感排名前5.jpg')
plt.show()
```



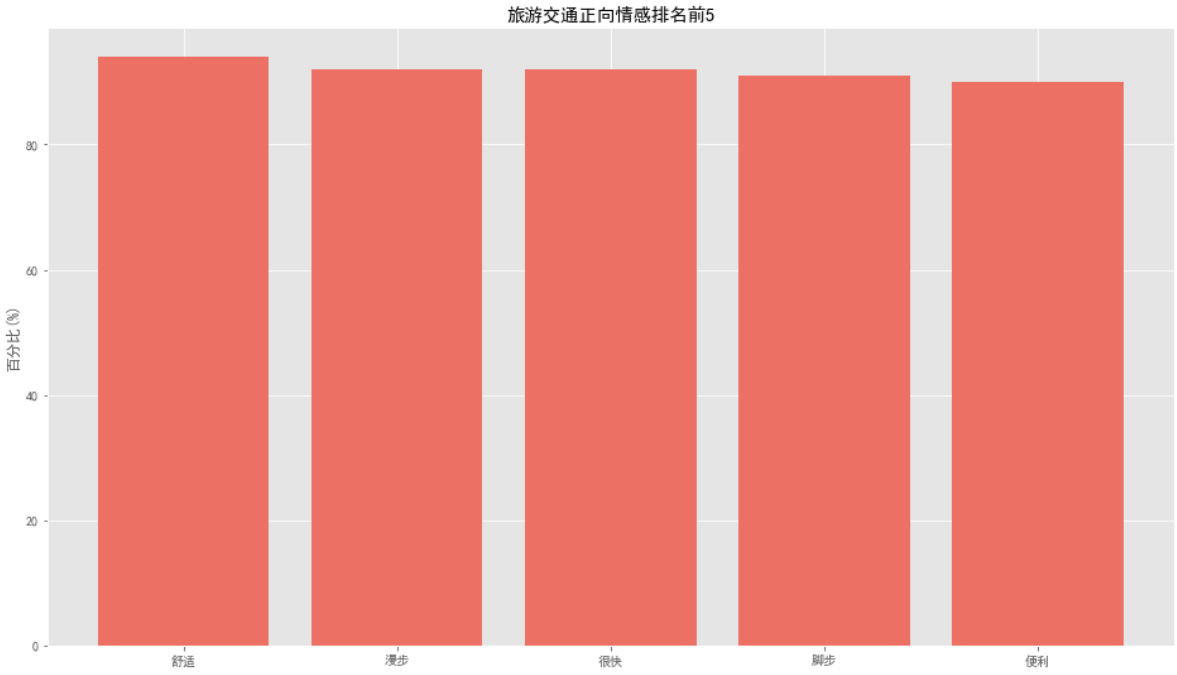
In [409]:

```
x_data11 = ['海枯石烂','黑灯瞎火','虫鸣','人生地不熟','浑浊']
y_data11 = [4.7,6,7.2,7.7,8]

plt.style.use('ggplot')
plt.figure(figsize=(16,9))
plt.bar(x_data11,y_data11,color='#707B7C')
plt.rcParams['font.sans-serif'] = ['SimHei']
plt.title("旅游吸引物正向情感倒数5名")
plt.ylabel("百分比(%)")
plt.savefig('旅游吸引物正向情感倒数5名.jpg')
plt.show()
```



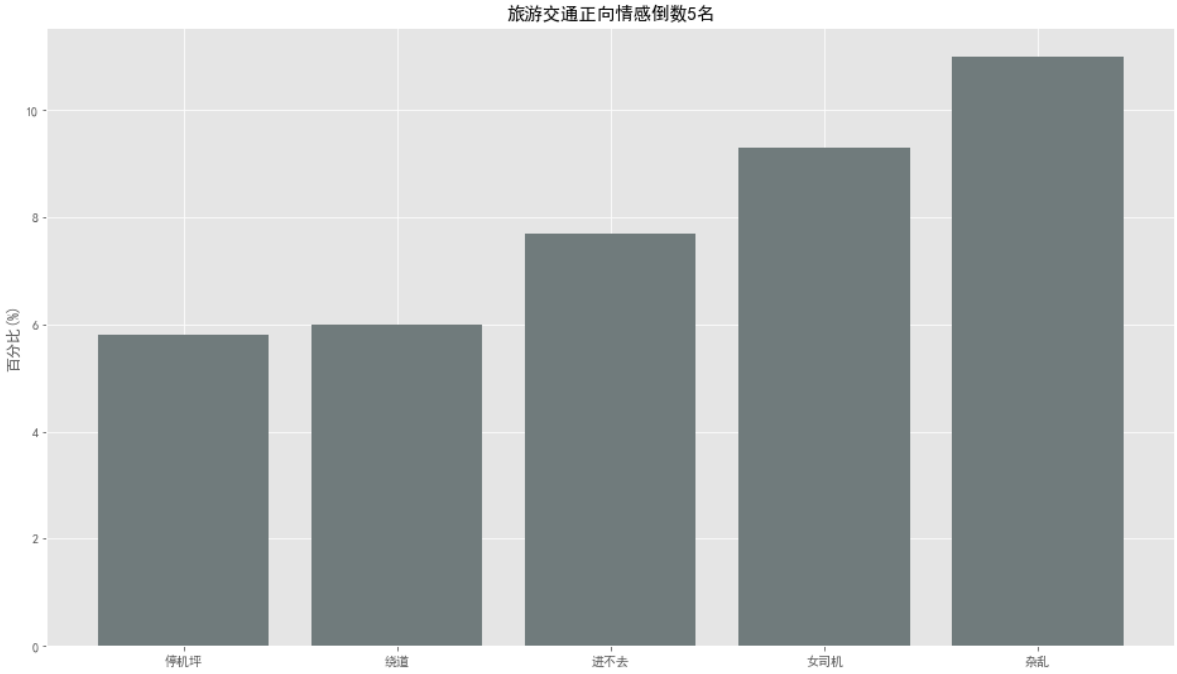
In [410]:

```
x_data12 = ['舒适','漫步','很快','脚步','便利']
y_data12 = [94,92,92,91,90]
plt.style.use('ggplot')
plt.figure(figsize=(16,9))
plt.bar(x_data12,y_data12,color='#EC7063')
plt.rcParams['font.sans-serif'] = ['SimHei']
plt.title("旅游交通正向情感排名前5")
plt.ylabel("百分比(%)")
plt.savefig('旅游交通正向情感排名前5.jpg')
plt.show()
```



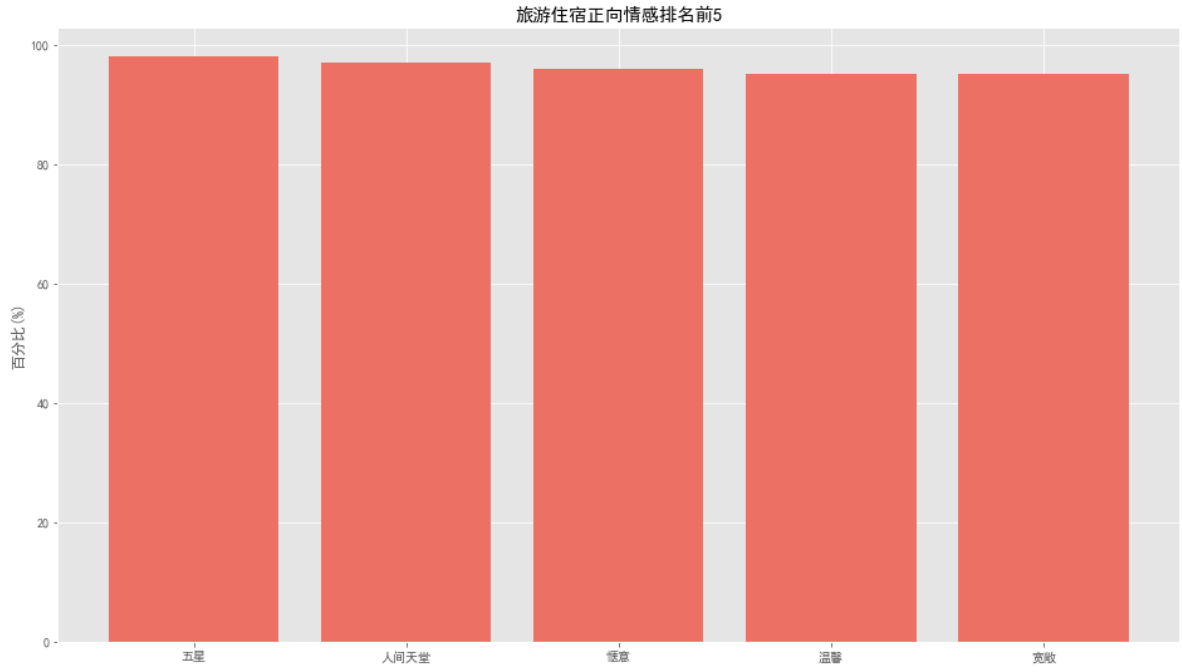
In [411]:

```
x_data13 = ['停机坪','绕道','进不去','女司机','杂乱']
y_data13 = [5.8,6,7.7,9.3,11]

plt.style.use('ggplot')
plt.figure(figsize=(16,9))
plt.bar(x_data13,y_data13,color='#707B7C')
plt.rcParams['font.sans-serif'] = ['SimHei']
plt.title("旅游交通正向情感倒数5名")
plt.ylabel("百分比(%)")
plt.savefig('旅游交通正向情感倒数5名.jpg')
plt.show()
```



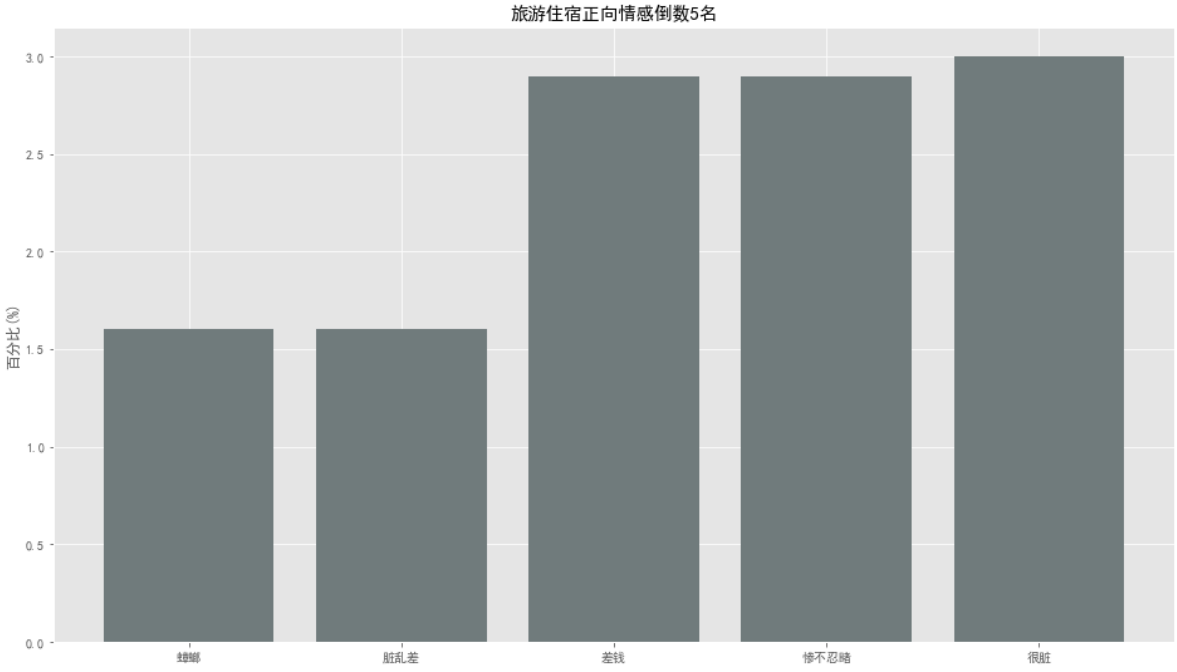
In [412]:

```
x_data14 = ['五星','人间天堂','惬意','温馨','宽敞']
y_data14 = [98,97,96,95,95]

plt.style.use('ggplot')
plt.figure(figsize=(16,9))
plt.bar(x_data14,y_data14,color='#EC7063')
plt.rcParams['font.sans-serif'] = ['SimHei']
plt.title("旅游住宿正向情感排名前5")
plt.ylabel("百分比(%)")
plt.savefig('旅游住宿正向情感排名前5.jpg')
plt.show()
```



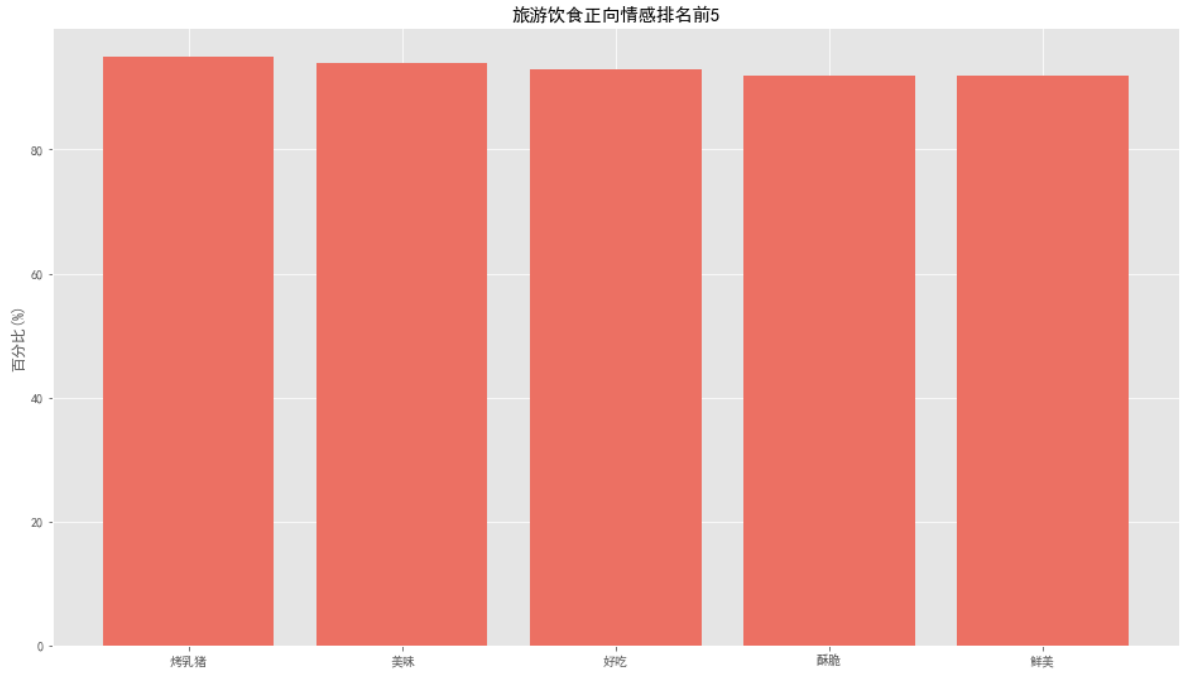
In [414]:

```
x_data15 = ['蟑螂','脏乱差','差钱','惨不忍睹','很脏']
y_data15 = [1.6,1.6,2.9,2.9,3]

plt.style.use('ggplot')
plt.figure(figsize=(16,9))
plt.bar(x_data15,y_data15,color='#707B7C')
plt.rcParams['font.sans-serif'] = ['SimHei']
plt.title("旅游住宿正向情感倒数5名")
plt.ylabel("百分比(%)")
plt.savefig('旅游住宿正向情感倒数5名.jpg')
plt.show()
```



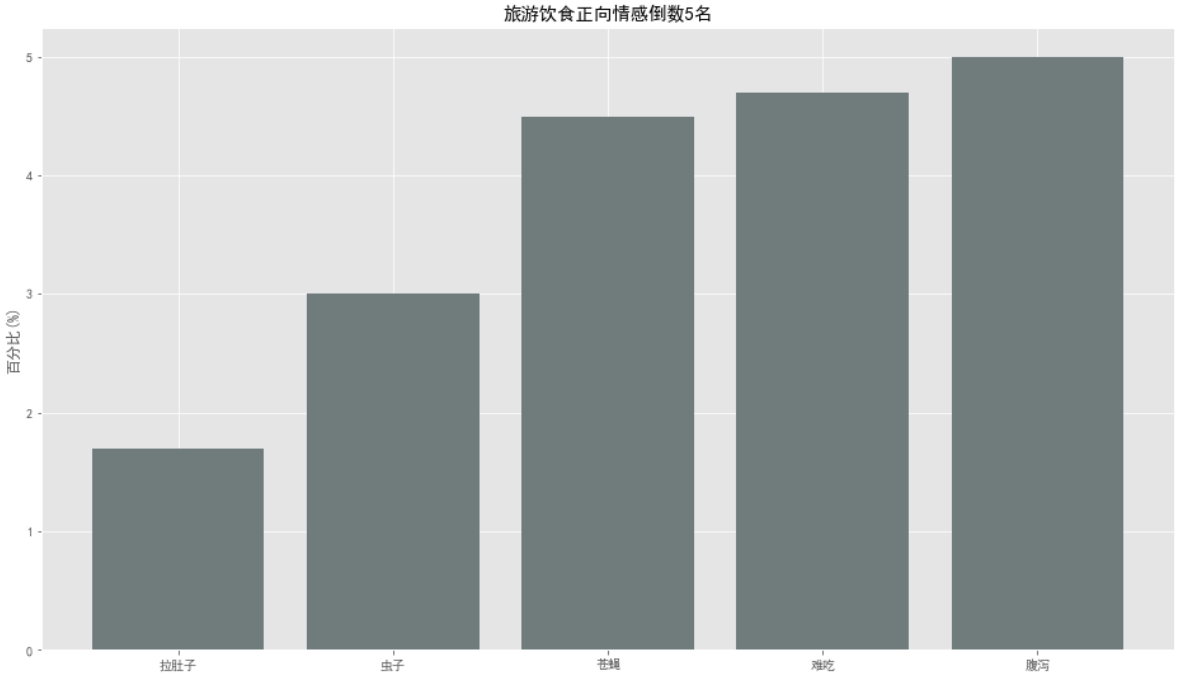
In [415]:

```
x_data16 = ['烤乳猪','美味','好吃','酥脆','鲜美']
y_data16 = [95,94,93,92,92]

plt.style.use('ggplot')
plt.figure(figsize=(16,9))
plt.bar(x_data16,y_data16,color='#EC7063')
plt.rcParams['font.sans-serif'] = ['SimHei']
plt.title("旅游饮食正向情感排名前5")
plt.ylabel("百分比(%)")
plt.savefig('旅游饮食正向情感排名前5.jpg')
plt.show()
```



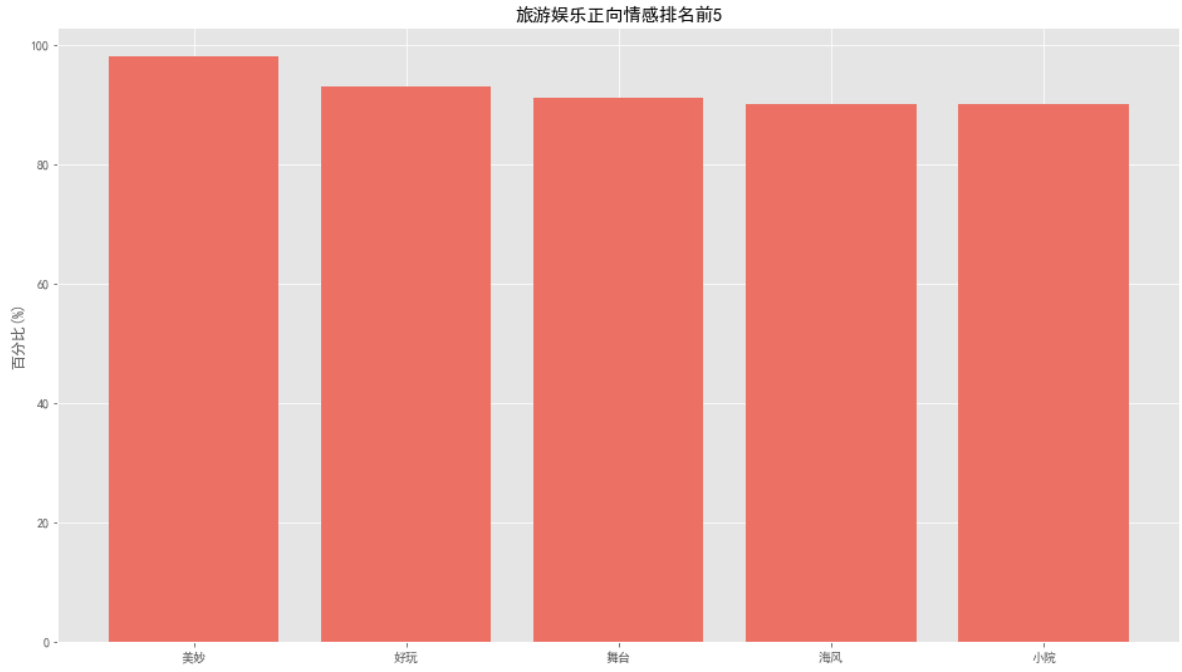
In [416]:

```
x_data17 = ['拉肚子','虫子','苍蝇','难吃','腹泻']
y_data17 = [1.7,3,4.5,4.7,5]

plt.style.use('ggplot')
plt.figure(figsize=(16,9))
plt.bar(x_data17,y_data17,color='#707B7C')
plt.rcParams['font.sans-serif'] = ['SimHei']
plt.title("旅游饮食正向情感倒数5名")
plt.ylabel("百分比(%)")
plt.savefig('旅游饮食正向情感倒数5名.jpg')
plt.show()
```



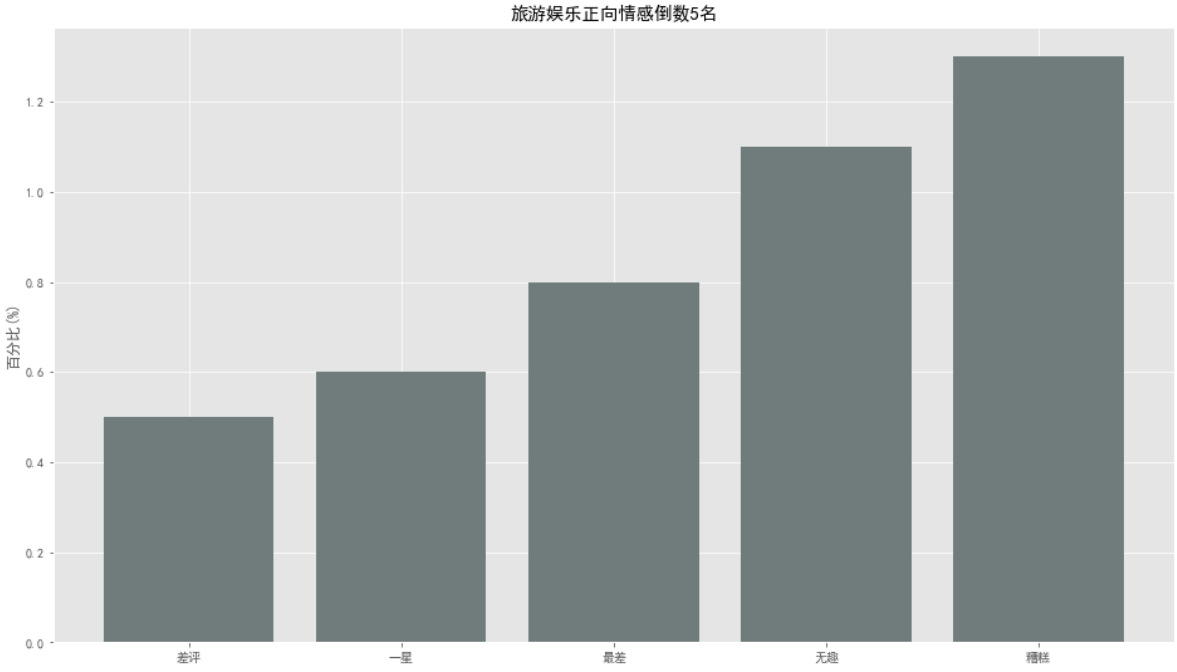
In [417]:

```
x_data18 = ['美妙','好玩','舞台','海风','小院']
y_data18 = [98,93,91,90,90]

plt.style.use('ggplot')
plt.figure(figsize=(16,9))
plt.bar(x_data18,y_data18,color='#EC7063')
plt.rcParams['font.sans-serif'] = ['SimHei']
plt.title("旅游娱乐正向情感排名前5")
plt.ylabel("百分比(%)")
plt.savefig('旅游娱乐正向情感排名前5.jpg')
plt.show()
```



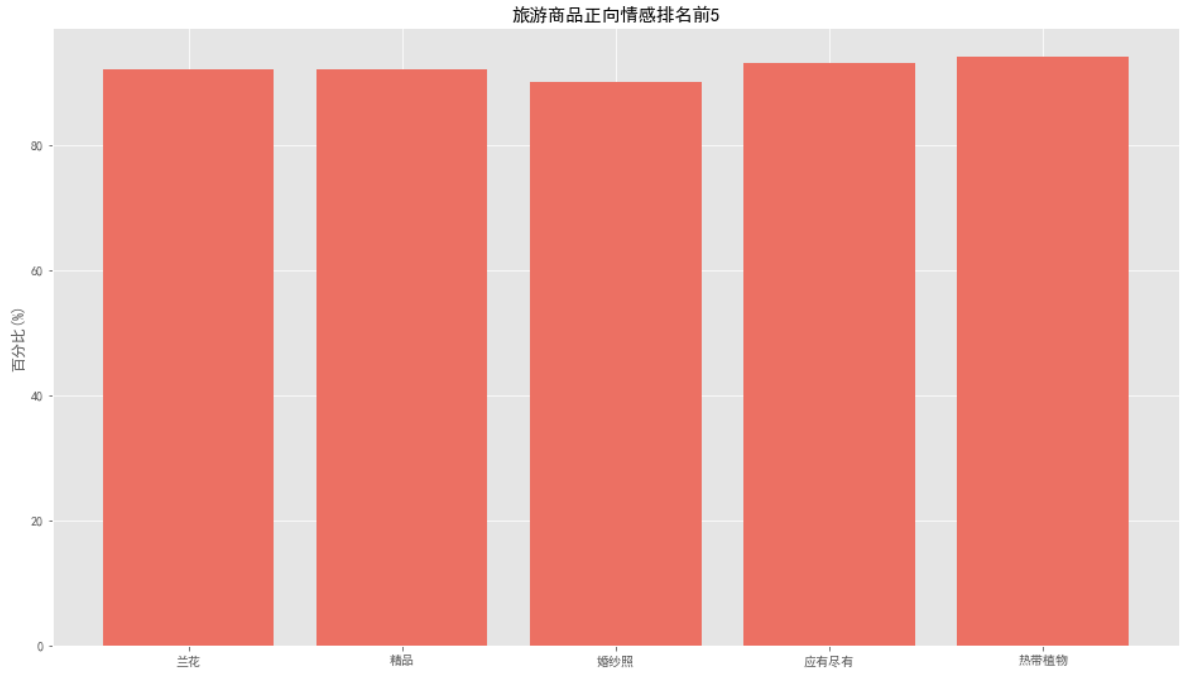
In [418]:

```
x_data19 = ['差评','一星','最差','无趣','糟糕']
y_data19 = [0.5,0.6,0.8,1.1,1.3]

plt.style.use('ggplot')
plt.figure(figsize=(16,9))
plt.bar(x_data19,y_data19,color='#707B7C')
plt.rcParams['font.sans-serif'] = ['SimHei']
plt.title("旅游娱乐正向情感倒数5名")
plt.ylabel("百分比(%)")
plt.savefig('旅游娱乐正向情感倒数5名.jpg')
plt.show()
```



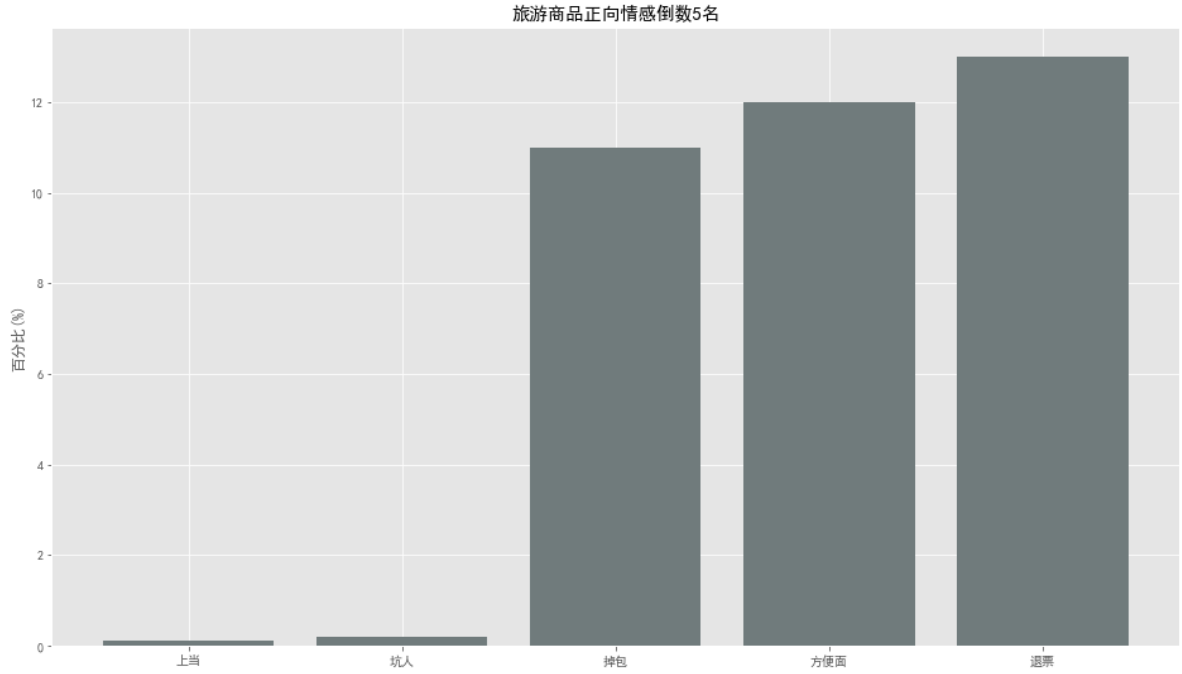
In [419]:

```
x_data20 = ['兰花','精品','婚纱照','应有尽有','热带植物']
y_data20 = [92,92,90,93,94]

plt.style.use('ggplot')
plt.figure(figsize=(16,9))
plt.bar(x_data20,y_data20,color='#EC7063')
plt.rcParams['font.sans-serif'] = ['SimHei']
plt.title("旅游商品正向情感排名前5")
plt.ylabel("百分比(%)")
plt.savefig('旅游商品正向情感排名前5.jpg')
plt.show()
```



In [420]:

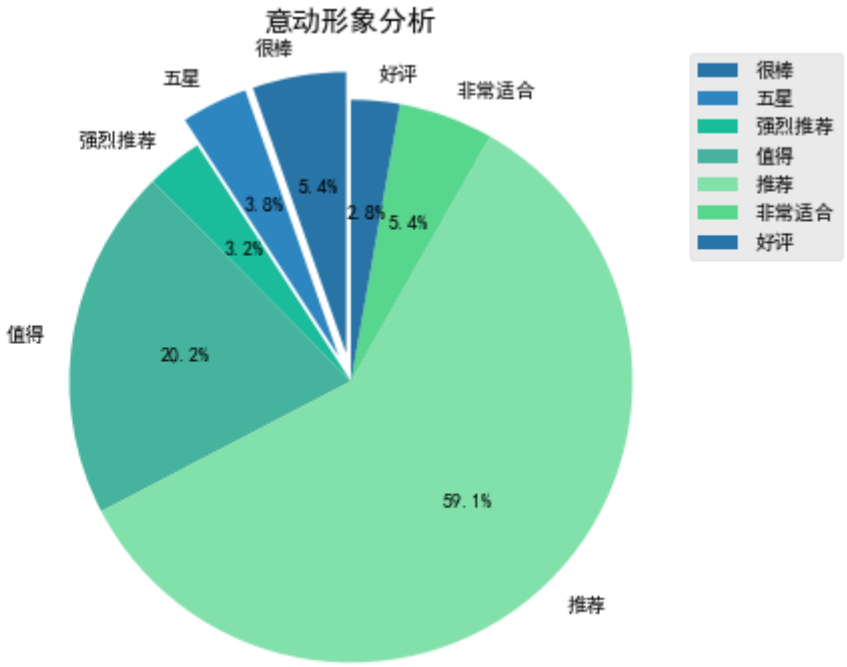
```
x_data21 = ['上当','坑人','掉包','方便面','退票']
y_data21 = [0.1,0.2,11,12,13]

plt.style.use('ggplot')
plt.figure(figsize=(16,9))
plt.bar(x_data21,y_data21,color='#707B7C')
plt.rcParams['font.sans-serif'] = ['SimHei']
plt.title("旅游商品正向情感倒数5名")
plt.ylabel("百分比(%)")
plt.savefig('旅游商品正向情感倒数5名.jpg')
plt.show()
```



上面的六个正面最5的柱状图是根据正面情感最高的高频词最其进行排序 正面倒数的六个柱状图则是根据正面情感最低的高频词的从情感分数最低的5个数进行排序画图

In [422]:

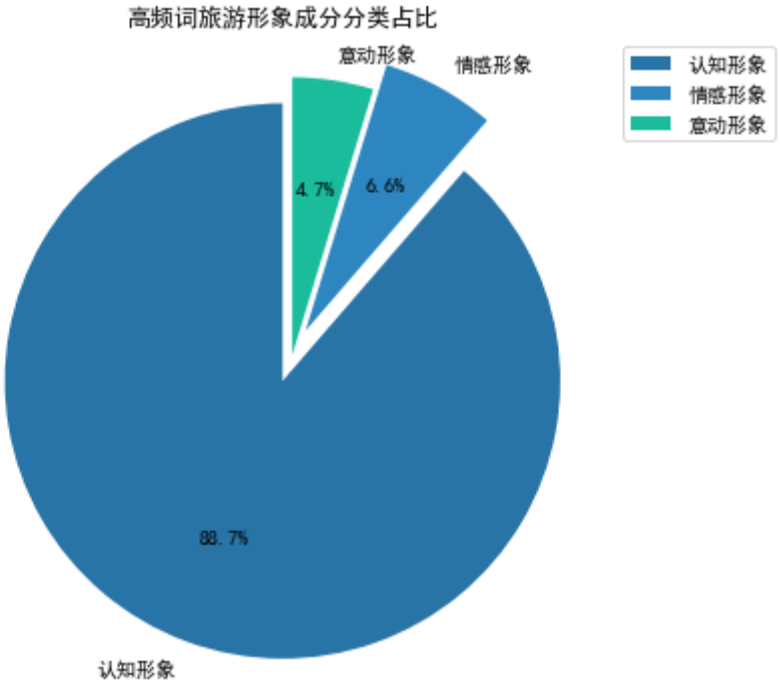
```
x_data22 = ['很棒','五星','强烈推荐','值得','推荐','非常适合','好评']
y_data22 = [241,171,145,908,2652,244,125]

plt.figure(figsize=(9,6)) #调节图形大小
labels = x_data22 #定义标签
sizes = y_data22 #每块值
colors = ['#2874A6','#2E86C1','#1ABC9C','#45B39D','#82E0AA','#58D68D'] #每块颜色定义
explode = (0.1,0.1,0,0,0,0,0) #将某一块分割出来，值越大分割出的间隙越大
patches,text1,text2 = plt.pie(sizes,
                      explode=explode,
                      labels=labels,
                      colors=colors,
                      labeldistance = 1.1,#图例距圆心半径倍距离
                      autopct = '%3.1f%%', #数值保留固定小数位
                      shadow = False, #无阴影设置
                      startangle =90, #逆时针起始角度设置
                      pctdistance = 0.6) #数值距圆心半径倍数距离
#patches饼图的返回值，texts1饼图外label的文本，texts2饼图内部文本
# x，y轴刻度设置一致，保证饼图为圆形
plt.axis('equal')
plt.title('意动形象分析')
plt.legend()
plt.savefig('意动形象分析.jpg')
plt.show()
```



该饼图是根据正面情感TOP100高频词的意动形象的统计归类画出的饼图

In [45]:

```
x_data22 = ['认知形象','情感形象','意动形象']
y_data22 = [84458,6313,4486]

plt.figure(figsize=(9,6)) #调节图形大小
labels = x_data22 #定义标签
sizes = y_data22 #每块值
colors = ['#2874A6','#2E86C1','#1ABC9C','#45B39D','#82E0AA','#58D68D'] #每块颜色定义
explode = (0.1,0.1,0) #将某一块分割出来，值越大分割出的间隙越大
patches,text1,text2 = plt.pie(sizes,
                      explode=explode,
                      labels=labels,
                      colors=colors,
                      labeldistance = 1.1,#图例距圆心半径倍距离
                      autopct = '%3.1f%%', #数值保留固定小数位
                      shadow = False, #无阴影设置
                      startangle =90, #逆时针起始角度设置
                      pctdistance = 0.6) #数值距圆心半径倍数距离
#patches饼图的返回值，texts1饼图外label的文本，texts2饼图内部文本
# x，y轴刻度设置一致，保证饼图为圆形
plt.axis('equal')
plt.title('高频词旅游形象成分分类占比')
plt.legend()
plt.savefig('高频词旅游形象成分分类占比.jpg')
plt.show()
```



这个则是根据高频词对词进行归类，然后找寻对应最高的前10的词

In [ ]:

```

```