In [12]:
```
#importing packages
from pandas import Series,  from numpy.random import randn
from statsmodels.stats.weightstats import ttest_ind
import scipy.stats as stats
from scipy.stats import ttest_ind
import pandas as pd
import numpy as np
from sklearn.impute import SimpleImputer
import seaborn as sb
%matplotlib inline
from sklearn.preprocessing import LabelEncoder
from matplotlib import pyplot as plt
from scipy.stats import shapiro
from scipy import stats
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error
from sklearn.metrics import classification_report
from sklearn.pipeline import Pipeline
from sklearn.preprocessing import StandardScaler
from sklearn.preprocessing import PolynomialFeatures
from sklearn.metrics import r2_score
from sklearn.model_selection import train_test_split
from sklearn.model_selection import cross_val_score
from sklearn.model_selection import cross_val_predict
from sklearn.linear_model import Ridge
from sklearn.model_selection import GridSearchCV
from datetime import datetime
from datetime import date
import statsmodels.api as sm
from statsmodels.formula.api import ols
from sklearn.preprocessing import StandardScaler
from sklearn.decomposition import PCA
from sklearn.linear_model import LogisticRegression as lgr
from mpl_toolkits.mplot3d import Axes3D
from sklearn.tree import DecisionTreeClassifier
from sklearn import preprocessing as preproc
from sklearn import metrics
from sklearn.datasets import make_classification
from sklearn.linear_model import LogisticRegression
from sklearn.model_selection import train_test_split
from sklearn.pipeline import make_pipeline
from sklearn.preprocessing import StandardScaler
from sklearn.neighbors import KNeighborsRegressor
from sklearn.neural_network import MLPRegressor
from sklearn.ensemble import RandomForestClassifier

from sklearn.metrics import accuracy_score, auc, balanced_accuracy_score, confusion_matrix, f1_score, precision
from sklearn.tree import DecisionTreeRegressor
from sklearn.linear_model import LogisticRegression
from sklearn.ensemble import RandomForestClassifier, IsolationForest, RandomForestClassifier
from sklearn.model_selection import cross_val_score, cross_val_predict, cross_validate, train_test_split
from sklearn.model_selection import GridSearchCV, RandomizedSearchCV

from sklearn.svm import SVC
```

In [13]:
```
zaub=pd.read_excel("C:\\Users\\sujoydutta\\Desktop\\Data analysis\\Datasets for ML\\Classifier\\zauberjoe fight
zaub.head()
```

Out[13]:

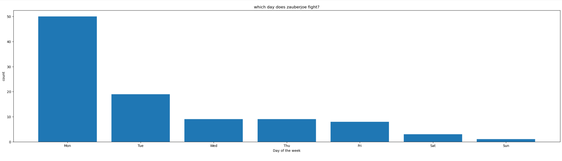
	Day	Time	Map	Winner	Loser	Victory?
0	Mon	20:13:54	MAP462	ZAUBERJOE	MAJORDAVE	1
1	Mon	21:13:54	MAP462	ZAUBERJOE	GUILLTAC	1
2	Mon	22:13:54	MAP456	ZAUBERJOE	MADRAIDER	1
3	Mon	23:13:54	MAP224	ZAUBERJOE	BONKERS57	1
4	Mon	12:13:54	MAP462	ZAUBERJOE	DEATHTRON	1

In [14]:
```
zaub.info()
```

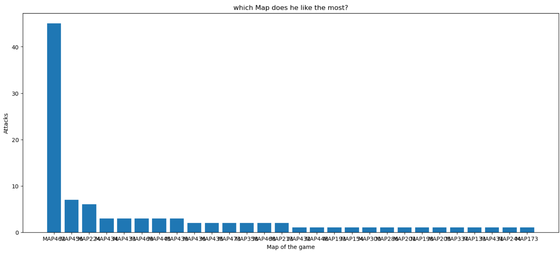
```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 99 entries, 0 to 98
Data columns (total 6 columns):
 #   Column    Non-Null Count  Dtype 
---  ------    --------------  ----- 
 0   Day       99 non-null     object
 1   Time      99 non-null     object
 2   Map       99 non-null     object
 3   Winner    99 non-null     object
 4   Loser     99 non-null     object
 5   Victory?  99 non-null     int64 
dtypes: int64(1), object(5)
memory usage: 4.8+ KB
```

In [15]:
```
zaub.Time=zaub.Time.astype(str)
```

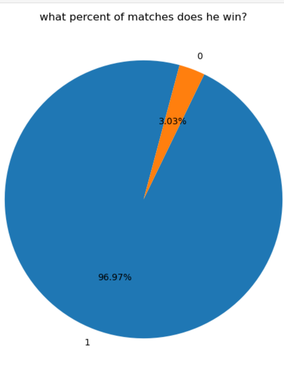
In [16]:
```
#which day does zauberjoe fight most?
day=zaub['Day'].unique()
count=zaub['Day'].value_counts()

plt.bar(day,count)


plt.title('which day does zauberjoe fight?')
plt.xlabel('Day of the week')
plt.ylabel('count')
plt.rcParams['figure.figsize'] = [17, 7]
plt.show()
```



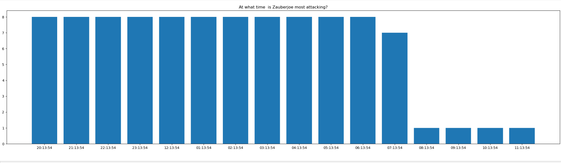
In [17]:
```
#which map does zauberjoe invade the most?
Map=zaub['Map'].unique()
count=zaub['Map'].value_counts()


plt.bar(Map,count)


plt.title('which Map does he like the most?')
plt.xlabel('Map of the game')
plt.ylabel('Attacks')
plt.rcParams['figure.figsize'] = [29,7]
plt.show()
```



In [18]:
```
#Victory percentage of Zauberjoe
vict=zaub['Victory?'].unique()
numbers=zaub['Victory?'].value_counts()

plt.title('what percent of matches does he win?')
plt.pie(numbers, labels = vict, startangle = 75,autopct='%1.2f%%')
plt.show()
```



In [19]:
```
time=zaub['Time']
x=time.unique()
y=time.value_counts()
plt.bar(x,y)
plt.title("At what time  is Zauberjoe most attacking?")
plt.rcParams['figure.figsize'] = [29,7]
```



In [22]:
```
#subset 462
zaub462=zaub[zaub['Map']=='MAP462']
```

In [24]:
```
zaub462
```

Out[24]:

	Day	Time	Map	Winner	Loser	Victory?
0	Mon	20:13:54	MAP462	ZAUBERJOE	MAJORDAVE	1
1	Mon	21:13:54	MAP462	ZAUBERJOE	GUILLTAC	1
4	Mon	12:13:54	MAP462	ZAUBERJOE	DEATHTRON	1
5	Mon	01:13:54	MAP462	ZAUBERJOE	MAJORDAVE	1
16	Tue	12:13:54	MAP462	ZAUBERJOE	MAJORDAVE	1
45	Tue	05:13:54	MAP462	ZAUBERJOE	DEATHTRON	1
78	Sat	02:13:54	MAP462	ZAUBERJOE	BR3	1

In [ ]:
```

```