In [45]:
```
import pandas as pd
import numpy as np
import sys
import seaborn as sns
import matplotlib.pyplot as plt
from sklearn.preprocessing import LabelEncoder, OneHotEncoder
from sklearn.preprocessing import MinMaxScaler, StandardScaler
```

In [14]:
```
default_col_labels = ["id", "Diagnosis"]
for i in range(1, 31):
    default_col_labels.append(f"F{i}")

data = pd.read_csv("data.csv", names=default_col_labels)
```

In [31]:
```
labelencoder = LabelEncoder()
y_encoded = labelencoder.fit_transform(data["Diagnosis"])
data["Diagnosis"] = y_encoded.reshape(-1, 1)
display(data)
```

	id	Diagnosis	F1	F2	F3	F4	F5	F6	F7	F8	...	F21	F22	F23
0	842302	1	17.99	10.38	122.80	1001.0	0.11840	0.27760	0.30010	0.14710	...	25.380	17.33	184.60
1	842517	1	20.57	17.77	132.90	1326.0	0.08474	0.07864	0.08690	0.07017	...	24.990	23.41	158.80
2	84300903	1	19.69	21.25	130.00	1203.0	0.10960	0.15990	0.19740	0.12790	...	23.570	25.53	152.50
3	84348301	1	11.42	20.38	77.58	386.1	0.14250	0.28390	0.24140	0.10520	...	14.910	26.50	98.87
4	84358402	1	20.29	14.34	135.10	1297.0	0.10030	0.13280	0.19800	0.10430	...	22.540	16.67	152.20
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
564	926424	1	21.56	22.39	142.00	1479.0	0.11100	0.11590	0.24390	0.13890	...	25.450	26.40	166.10
565	926682	1	20.13	28.25	131.20	1261.0	0.09780	0.10340	0.14400	0.09791	...	23.690	38.25	155.00
566	926954	1	16.60	28.08	108.30	858.1	0.08455	0.10230	0.09251	0.05302	...	18.980	34.12	126.70
567	927241	1	20.60	29.33	140.10	1265.0	0.11780	0.27700	0.35140	0.15200	...	25.740	39.42	184.60
568	92751	0	7.76	24.54	47.92	181.0	0.05263	0.04362	0.00000	0.00000	...	9.456	30.37	59.16

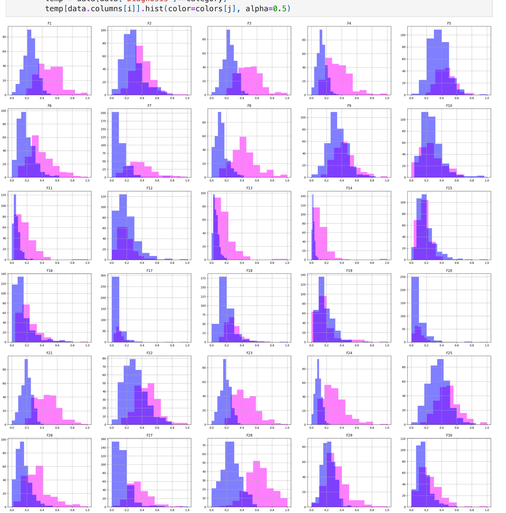
569 rows × 32 columns

In [55]:
```
#normalize the data
for i in range(2, len(data.columns)):
    minmaxscaler = MinMaxScaler()
    data[data.columns[i]] = minmaxscaler.fit_transform(data[data.columns[i]].values.reshape(-1,1))
display(data)
```

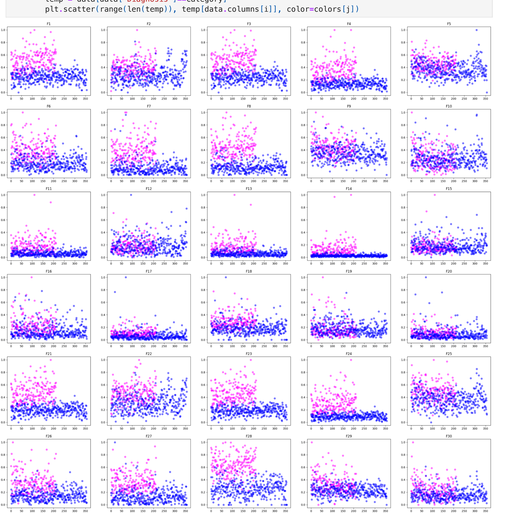
	id	Diagnosis	F1	F2	F3	F4	F5	F6	F7	F8	...
0	842302	1	0.521037	0.022658	0.545989	0.363733	0.593753	0.792037	0.703140	0.731113	...
1	842517	1	0.643144	0.272574	0.615783	0.501591	0.289880	0.181768	0.203608	0.348757	...
2	84300903	1	0.601496	0.390260	0.595743	0.449417	0.514309	0.431017	0.462512	0.635686	...
3	84348301	1	0.210090	0.360839	0.233501	0.102906	0.811321	0.811361	0.565604	0.522863	...
4	84358402	1	0.629893	0.156578	0.630986	0.489290	0.430351	0.347893	0.463918	0.518390	...
...	...	...	...	...	...	...	...	...	...	...	...
564	926424	1	0.690000	0.428813	0.678668	0.566490	0.526948	0.296055	0.571462	0.690358	...
565	926682	1	0.622320	0.626987	0.604036	0.474019	0.407782	0.257714	0.337395	0.486630	...
566	926954	1	0.455251	0.621238	0.445788	0.303118	0.288165	0.254340	0.216753	0.263519	...
567	927241	1	0.644564	0.663510	0.665538	0.475716	0.588336	0.790197	0.823336	0.755467	...
568	92751	0	0.036869	0.501522	0.028540	0.015907	0.000000	0.074351	0.000000	0.000000	...

569 rows × 32 columns

In [56]:
```
#Histogram of the data distribution after normalization
categories = data["Diagnosis"].unique()
colors = ["magenta", "blue"]

plt.figure(figsize=(31,31))

for i in range(2, len(data.columns)):
    plt.subplot(6, 5, i - 1)
    plt.title(data.columns[i])
    for j, category in enumerate(categories):
        temp = data[data["Diagnosis"]==category]
        temp[data.columns[i]].hist(color=colors[j], alpha=0.5)
```



In [75]:
```
#Scatter plot of the data distribution after normalization
categories = data["Diagnosis"].unique()
colors = ["magenta", "blue"]

plt.figure(figsize=(31,31))

for i in range(2, len(data.columns)):
    plt.subplot(6, 5, i - 1)
    plt.title(data.columns[i])
    for j, category in enumerate(categories):
        temp = data[data["Diagnosis"]==category]
        plt.scatter(range(len(temp)), temp[data.columns[i]], color=colors[j])
```

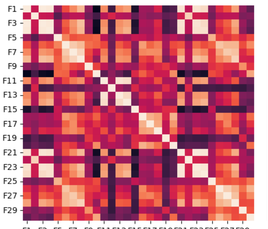


In [72]:
```
y = data["Diagnosis"]
x = data.iloc[:,2:]
display(x)
```

	F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	...	F2
0	0.521037	0.022658	0.545989	0.363733	0.593753	0.792037	0.703140	0.731113	0.686364	0.605518	...	0.6207
1	0.643144	0.272574	0.615783	0.501591	0.289880	0.181768	0.203608	0.348757	0.379798	0.141323	...	0.6069
2	0.601496	0.390260	0.595743	0.449417	0.514309	0.431017	0.462512	0.635686	0.509596	0.211247	...	0.5563
3	0.210090	0.360839	0.233501	0.102906	0.811321	0.811361	0.565604	0.522863	0.776263	1.000000	...	0.2483
4	0.629893	0.156578	0.630986	0.489290	0.430351	0.347893	0.463918	0.518390	0.378283	0.186816	...	0.5197
...	...	...	...	...	...	...	...	...	...	...	...	...
564	0.690000	0.428813	0.678668	0.566490	0.526948	0.296055	0.571462	0.690358	0.336364	0.132056	...	0.6232
565	0.622320	0.626987	0.604036	0.474019	0.407782	0.257714	0.337395	0.486630	0.349495	0.113100	...	0.5606
566	0.455251	0.621238	0.445788	0.303118	0.288165	0.254340	0.216753	0.263519	0.267677	0.137321	...	0.3930
567	0.644564	0.663510	0.665538	0.475716	0.588336	0.790197	0.823336	0.755467	0.675253	0.425442	...	0.6335
568	0.036869	0.501522	0.028540	0.015907	0.000000	0.074351	0.000000	0.000000	0.266162	0.187026	...	0.0542

569 rows × 30 columns

Out[72]:
```
<Axes: >
```



In [ ]:
```
```