In [1]:
```
#we can do column Normalization using 4 methods
#1.using maximum absolute scaling
#2.Using the Min Max Feature Scaling
#3.Using the Z-Score Method
#4.Using Sklearn
```

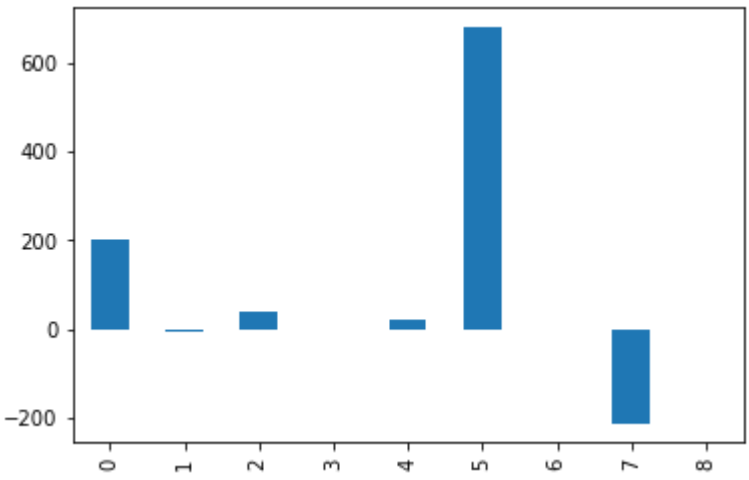
In [7]:
```
#First Creating a Dataframe
import pandas as pd

#create data
df=pd.DataFrame({'Column 1':[200,-4,39,0.5,22.3,680.1,0.005,-211.3,-0.12],
                 'Column 2':[20,30,40.2,63.89,59.42,0.05,-29.32,92.1,-4.9],
                 'Column 3':[10,20,40,29,52,11,67,82,40]})
```

In [8]:
```
display(df)
```

	Column 1	Column 2	Column 3
0	200.000	20.00	10
1	-4.000	30.00	20
2	39.000	40.20	40
3	0.500	63.89	29
4	22.300	59.42	52
5	680.100	0.05	11
6	0.005	-29.32	67
7	-211.300	92.10	82
8	-0.120	-4.90	40

In [9]:
```
#As we see the  column 1 and 2 is not normalized and column 3 is Normalized.
df['Column 1'].plot(kind='bar')
```

Out[9]: <AxesSubplot:>



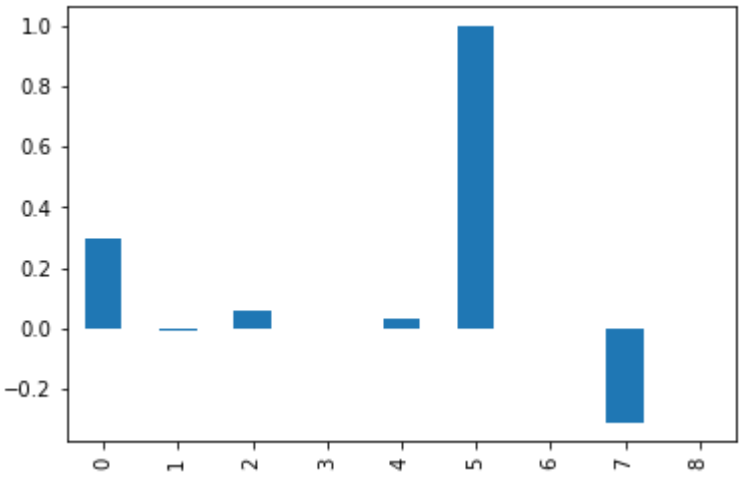
Using Maximum absolute Scaling

In [13]:
```
#copy the Data
df_max_scaled=df.copy(deep=True)

#apply Normalization Technique on Column 1
column='Column 1'
df_max_scaled[column]=df_max_scaled[column]/df_max_scaled[column].abs().max()

#view normalized data
display(df_max_scaled)
```

	Column 1	Column 2	Column 3
0	0.294074	20.00	10
1	-0.005881	30.00	20
2	0.057345	40.20	40
3	0.000735	63.89	29
4	0.032789	59.42	52
5	1.000000	0.05	11
6	0.000007	-29.32	67
7	-0.310690	92.10	82
8	-0.000176	-4.90	40

In [14]:
```
df_max_scaled[column].plot(kind='bar')
```

Out[14]: <AxesSubplot:>



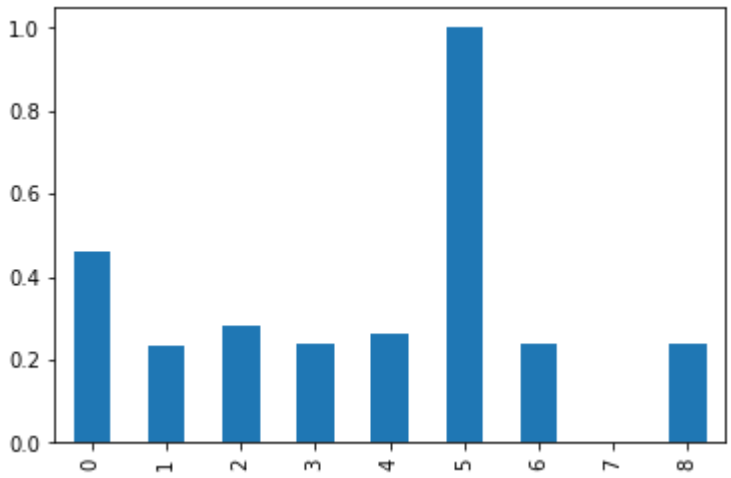
Using Min Max feature scaling

In [20]:
```
import pandas as pd
df_max_scaled=df.copy()

column='Column 1'
df_max_scaled[column]=(df_max_scaled[column]-df_max_scaled[column].min())/(df_ma

#Displaying the plot
display(df_max_scaled)
```

	Column 1	Column 2	Column 3
0	0.461409	20.00	10
1	0.232556	30.00	20
2	0.280794	40.20	40
3	0.237604	63.89	29
4	0.262060	59.42	52
5	1.000000	0.05	11
6	0.237048	-29.32	67
7	0.000000	92.10	82
8	0.236908	-4.90	40

In [21]:
```
df_max_scaled[column].plot(kind='bar')
```

Out[21]: <AxesSubplot:>



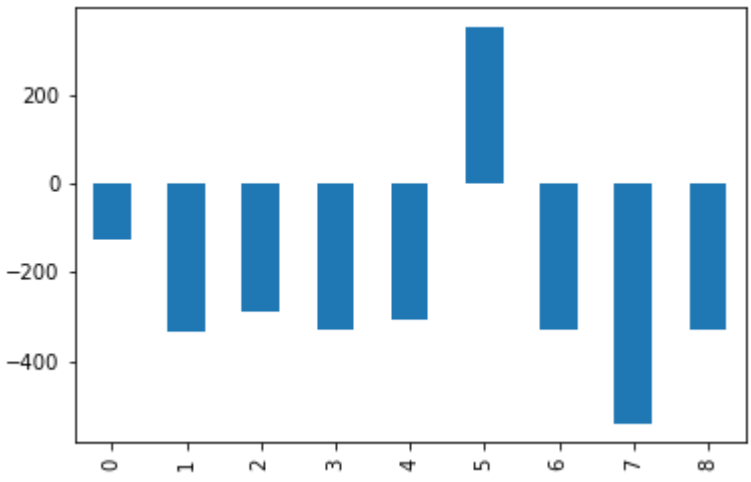
using the Z-Score Method

In [25]:
```
df_max_scaled=df.copy()

column='Column 1'
df_max_scaled[column]=(df_max_scaled[column]-df_max_scaled[column].mean())-(df_ma

display(df_max_scaled)
```

	Column 1	Column 2	Column 3
0	-128.353344	20.00	10
1	-332.353344	30.00	20
2	-289.353344	40.20	40
3	-327.853344	63.89	29
4	-306.053344	59.42	52
5	351.746656	0.05	11
6	-328.348344	-29.32	67
7	-539.653344	92.10	82
8	-328.473344	-4.90	40

In [26]:
```
df_max_scaled[column].plot(kind='bar')
```

Out[26]: <AxesSubplot:>



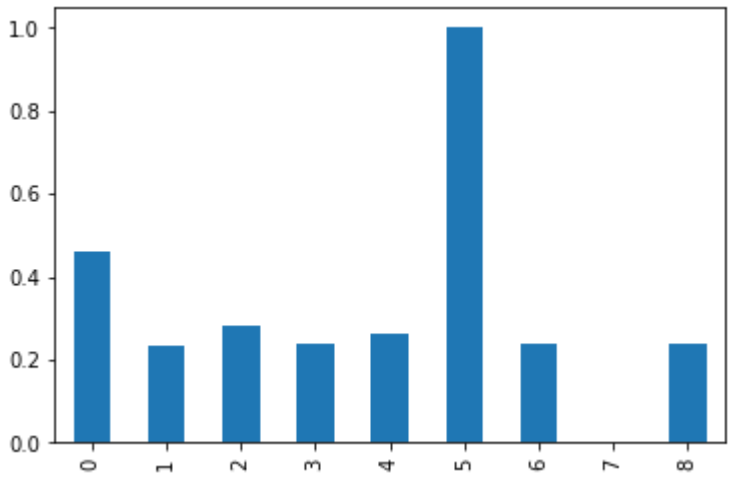
Using Sklearn

In [32]:
```
from sklearn.preprocessing import MinMaxScaler
import numpy as np

df=df.copy()

column='Column 1'
df[column]=MinMaxScaler().fit_transform(np.array(df[column]).reshape(-1,1))

display(df)
```

	Column 1	Column 2	Column 3
0	0.461409	20.00	10
1	0.232556	30.00	20
2	0.280794	40.20	40
3	0.237604	63.89	29
4	0.262060	59.42	52
5	1.000000	0.05	11
6	0.237048	-29.32	67
7	0.000000	92.10	82
8	0.236908	-4.90	40

In [34]:
```
df[column].plot(kind='bar')
```

Out[34]: <AxesSubplot:>



In [ ]:
```
#compared to all min max scaler method and Sklearn method's are most efficient.
```