In [8]:
```
import pandas as pd
import numpy as np
```

Q.no.1

In [9]:
```
df = pd.read_csv('NCI60_data.csv')
df = df.set_index('Unnamed: 0')
df_labels = pd.read_csv('NCI60_labs.csv')
df_labels = df_labels.set_index('Unnamed: 0')
```

In [10]:
```
df.head(5)
```

Out[10]:

	1	2	3	4	5	6	7	8
Unnamed: 0								
V1	0.300000	1.180000	0.550000	1.140000	-0.265000	-7.000000e-02	0.350000	-0.315000
V2	0.679961	1.289961	0.169961	0.379961	0.464961	5.799610e-01	0.699961	0.724961
V3	0.940000	-0.040000	-0.170000	-0.040000	-0.605000	0.000000e+00	0.090000	0.645000
V4	0.280000	-0.310000	0.680000	-0.810000	0.625000	-1.387779e-17	0.170000	0.245000
V5	0.485000	-0.465000	0.395000	0.905000	0.200000	-5.000000e-03	0.085000	0.110000

5 rows × 6830 columns

In [11]:
```
df_labels.head(5)
```

Out[11]:

	x
Unnamed: 0	
1	CNS
2	CNS
3	CNS
4	RENAL
5	BREAST

Q.no.2

In [12]:
```
df.describe()
```

Out[12]:
```
              1          2          3          4          5          6          7          8
count 64.000000  64.000000  64.000000  64.000000  64.000000  64.000000  64.000000  64.000000  64.
 mean -0.019063  -0.027813  -0.019923  -0.328673   0.026093   0.006718   0.019687  -0.023126   0.
  std  0.441332   0.757433   0.433306   1.091905   0.485073   0.350432   0.370683   0.338629   0.
  min -1.060000  -2.190000  -1.710000  -2.610000  -0.825000  -0.700000  -0.920000  -0.705000  -0.
  25% -0.372500  -0.404985  -0.192485  -1.322500  -0.225000  -0.156250  -0.246250  -0.204985  -0.
  50%  0.000000   0.000000   0.000000   0.000000   0.000000   0.000000   0.000000   0.000000   0.
  75%  0.310005   0.352500   0.162490   0.692500   0.210000   0.184995   0.247505   0.160015   0.
  max  0.940000   2.240000   1.150000   1.500000   1.715000   1.160000   0.940000   0.724961   0.

8 rows × 6830 columns
```

It's clearly evident from above that the numerical values in the dataset are not standardized as neither the mean is 0 for these columns nor is the standard deviation 1. In, other words, the columns have different means and standard deviations, which means that they are not standardised. So let's standardize them:

In [13]:
```
from sklearn import preprocessing
df.loc[:] = preprocessing.scale(df.loc[:], axis=0)
```

In [14]:
```
df.describe()
```

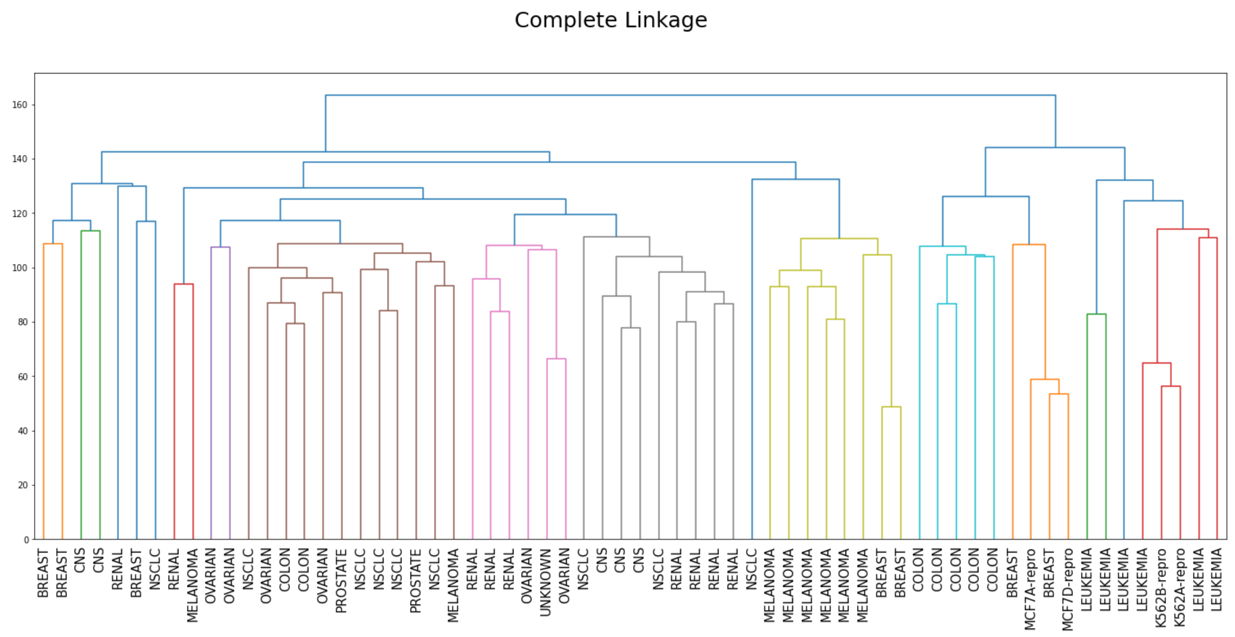
Out[14]:
```
                  1          2              3              4              5              6
count  6.400000e+01  64.000000   6.400000e+01   6.400000e+01   6.400000e+01   6.400000e+01  6.40
 mean -8.673617e-18   0.000000  -3.014082e-17   3.989864e-17  -2.428613e-17  -5.204170e-18  2.42
  std  1.007905e+00   1.007905   1.007905e+00   1.007905e+00   1.007905e+00   1.007905e+00  1.00
  min -2.377270e+00  -2.877193  -3.931262e+00  -2.105826e+00  -1.768435e+00  -2.032645e+00  -2.55
  25% -8.071713e-01  -0.501898  -4.013951e-01  -9.173725e-01  -5.217310e-01  -4.687243e-01  -7.23
  50%  4.353664e-02   0.037011   4.634208e-02   3.033881e-01  -5.421676e-02  -1.932168e-02  -5.35
  75%  7.515195e-01   0.506077   4.243081e-01   9.426144e-01   3.821298e-01   5.127569e-01  6.19
  max  2.190290e+00   3.017748   2.721339e+00   1.687994e+00   3.509280e+00   3.317043e+00  2.50

8 rows × 6830 columns
```

Q.no.3 and Q.no.4

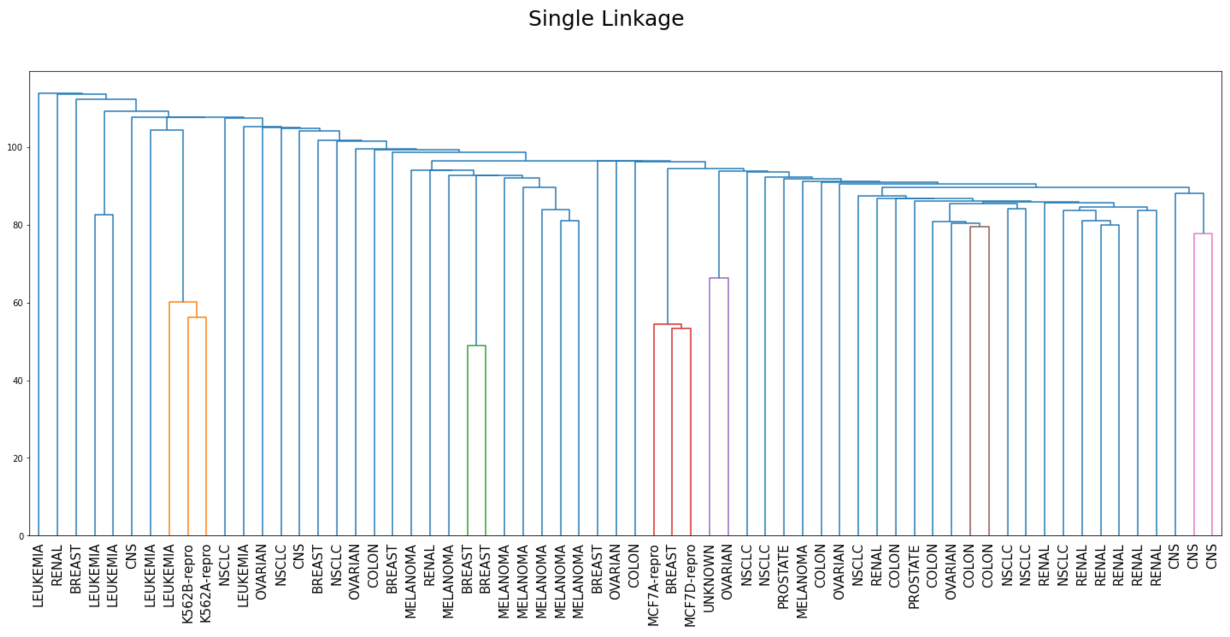
In [15]:
```
import matplotlib.pyplot as plt
from scipy.cluster.hierarchy import dendrogram, linkage
```

a.

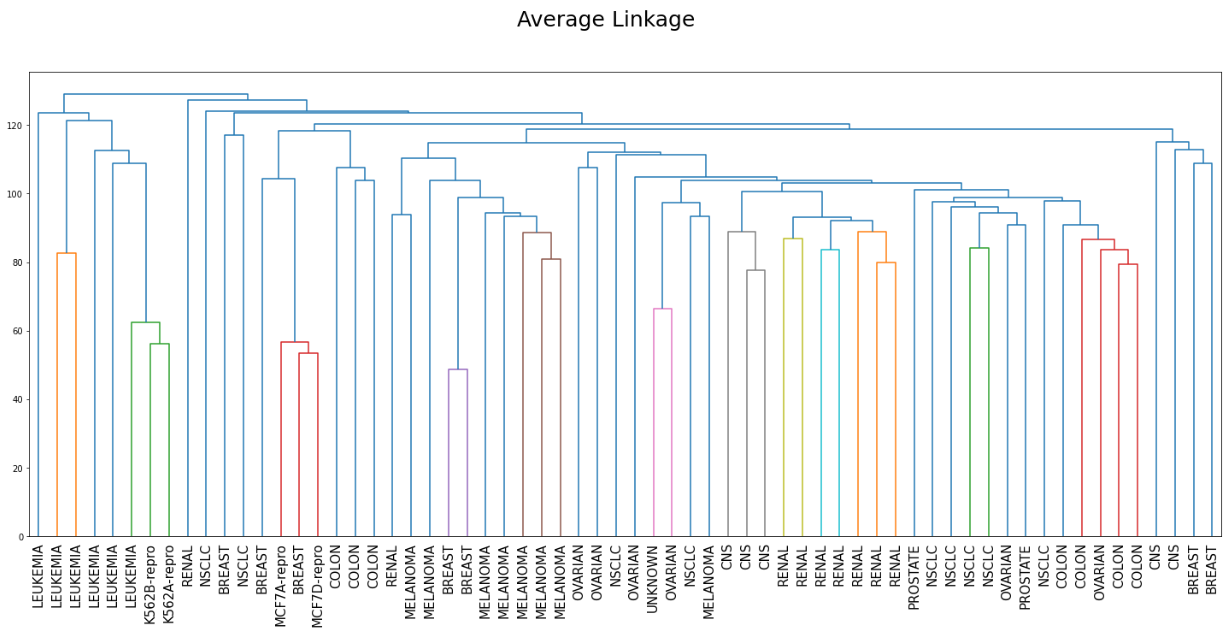
In [16]:
```
Z1= linkage(df, 'complete')
fig = plt.figure(figsize=(25, 10))
fig.suptitle("Complete Linkage", size=25)
dn = dendrogram(Z1, leaf_font_size = 15, labels = df_labels.x.values.tolist()) #
plt.show()
```



In [17]:
```
Z2= linkage(df, 'single')
fig = plt.figure(figsize=(25, 10))
fig.suptitle("Single Linkage", size=25)
dn = dendrogram(Z2, leaf_font_size = 15, labels = df_labels.x.values.tolist()) #
plt.show()
```



In [18]:
```
Z3= linkage(df, 'average')
fig = plt.figure(figsize=(25, 10))
fig.suptitle("Average Linkage", size=25)
dn = dendrogram(Z3, leaf_font_size = 15, labels = df_labels.x.values.tolist()) #
plt.show()
```



The dendrograms obtained from single linkage has extended clusters to which single leaves are fused one by one while the same obtained from complete and average linkages have evenly sized clusters and appear more balanced. Even comparing between clusters from complete and average linkage, the former (clusters obtained from complete linkage) are more balanced than the latter.

Moreover, the clusters obtained from complete linkage have a maximum distance of 160 (the greatest) while as the same obtained from average and single linkage have around 130 and 115 (the least) respectively. This might be the case because complete linkage takes maximum distance during the intermediate phases and single linkage takes minimum distance during the intermediate phases.

b.

The linkage that produced the best result is Complete Linkage as cancer types that are similar are grouped at smaller heights/distances as a result of which it has distinct colour codes within its subclusters and hence the clustering is easily interpretable and accurately employable.

c.

The number of clusters obtained at a cutoff distance of 139 is 4.

d.

In [20]:
```
from scipy.cluster.hierarchy import fcluster
column_labels = fcluster(Z1, t=4, criterion='maxclust')
pd.crosstab(df_labels.x, column_labels)
```

Out[20]:

col_0	1	2	3	4
x				
BREAST	3	2	2	0
CNS	2	3	0	0
COLON	0	2	5	0
K562A-repro	0	0	0	1
K562B-repro	0	0	0	1
LEUKEMIA	0	0	0	6
MCF7A-repro	0	0	1	0
MCF7D-repro	0	0	1	0
MELANOMA	0	8	0	0
NSCLC	1	8	0	0
OVARIAN	0	6	0	0
PROSTATE	0	2	0	0
RENAL	1	8	0	0
UNKNOWN	0	1	0	0

Q.no.5

In [21]:
```
from sklearn.cluster import KMeans
```

a.

In [22]:
```
km1 = KMeans(n_clusters=4, n_init=150, random_state=123)
km1.fit(df)
```

Out[22]:
```
KMeans(n_clusters=4, n_init=150, random_state=123)
```

b.

In [24]:
```
pd.crosstab(df.index, km1.labels_)
```

Out[24]:

col_0	0	1	2	3
row_0				
BREAST	2	2	3	0
CNS	0	0	5	0
COLON	0	7	0	0
K562A-repro	0	0	0	1
K562B-repro	0	0	0	1
LEUKEMIA	0	0	0	6
MCF7A-repro	0	1	0	0
MCF7D-repro	0	1	0	0
MELANOMA	7	0	1	0
NSCLC	0	3	6	0
OVARIAN	0	4	2	0
PROSTATE	0	1	1	0
RENAL	0	0	9	0
UNKNOWN	0	0	1	0

K-Means clustering appears to be more efficient in clustering the cancer types such as CNS, COLON, and RENAL while the hierarchial clustering appears to be more efficient in grouping cancers of type MELANOMA, NSLC, OVARIAN, and PROSTATE. The efficiency was based on the ability to group cancer of any given type into the same cluster (which was analyzed by counting the number of 0s in any row in the contingency table).