K- Means Algorithm

In [2]:
```
import warnings
warnings.filterwarnings('ignore')
```

In [3]:
```
# importing libraries

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

In [4]:
```
# importing dataset

dataset=pd.read_csv(r"C:\Users\Jan Saida\OneDrive\Documents\Desktop\Excel sheets\Mall_Customers.csv")
dataset
```

Out[4]:

	CustomerID	Genre	Age	Annual Income (k$)	Spending Score (1-100)
0	1	Male	19	15	39
1	2	Male	21	15	81
2	3	Female	20	16	6
3	4	Female	23	16	77
4	5	Female	31	17	40
...	...	...	...	...	...
195	196	Female	35	120	79
196	197	Female	45	126	28
197	198	Male	32	126	74
198	199	Male	32	137	18
199	200	Male	30	137	83

200 rows × 5 columns

In [5]:
```
x=dataset.iloc[:,[3,4]].values
x
```

Out[5]:
```
array([[ 15,  39],
       [ 15,  81],
       [ 16,   6],
       [ 16,  77],
       [ 17,  40],
       [ 17,  76],
       [ 18,   6],
       [ 18,  94],
       [ 19,   3],
       [ 19,  72],
       [ 19,  14],
       [ 19,  99],
       [ 20,  15],
       [ 20,  77],
       [ 20,  13],
       [ 20,  79],
       [ 21,  35],
       [ 21,  66],
       [ 23,  29],
       [ 23,  98],
       [ 24,  35],
       [ 24,  73],
       [ 25,   5],
       [ 25,  73],
       [ 28,  14],
       [ 28,  82],
       [ 28,  32],
       [ 28,  61],
       [ 29,  31],
       [ 29,  87],
       [ 30,   4],
       [ 30,  73],
       [ 33,   4],
       [ 33,  92],
       [ 33,  14],
       [ 33,  81],
       [ 34,  17],
       [ 34,  73],
       [ 37,  26],
       [ 37,  75],
       [ 38,  35],
       [ 38,  92],
       [ 39,  36],
       [ 39,  61],
       [ 39,  28],
```

```
[ 39,  65],
[ 40,  55],
[ 40,  47],
[ 40,  42],
[ 40,  42],
[ 42,  52],
[ 42,  60],
[ 43,  54],
[ 43,  60],
[ 43,  45],
[ 43,  41],
[ 44,  50],
[ 44,  46],
[ 46,  51],
[ 46,  46],
[ 46,  56],
[ 46,  55],
[ 47,  52],
[ 47,  59],
[ 48,  51],
[ 48,  59],
[ 48,  50],
[ 48,  48],
[ 48,  59],
[ 48,  47],
[ 49,  55],
[ 49,  42],
[ 50,  49],
[ 50,  56],
[ 54,  47],
[ 54,  54],
[ 54,  53],
[ 54,  48],
[ 54,  52],
[ 54,  42],
[ 54,  51],
[ 54,  55],
[ 54,  41],
[ 54,  44],
[ 54,  57],
[ 54,  46],
[ 57,  58],
[ 57,  55],
[ 58,  60],
[ 58,  46],
```

```
[ 59,  55],
[ 59,  41],
[ 60,  49],
[ 60,  40],
[ 60,  42],
[ 60,  52],
[ 60,  47],
[ 60,  50],
[ 61,  42],
[ 61,  49],
[ 62,  41],
[ 62,  48],
[ 62,  59],
[ 62,  55],
[ 62,  56],
[ 62,  42],
[ 63,  50],
[ 63,  46],
[ 63,  43],
[ 63,  48],
[ 63,  52],
[ 63,  54],
[ 64,  42],
[ 64,  46],
[ 65,  48],
[ 65,  50],
[ 65,  43],
[ 65,  59],
[ 67,  43],
[ 67,  57],
[ 67,  56],
[ 67,  40],
[ 69,  58],
[ 69,  91],
[ 70,  29],
[ 70,  77],
[ 71,  35],
[ 71,  95],
[ 71,  11],
[ 71,  75],
[ 71,   9],
[ 71,  75],
[ 72,  34],
[ 72,  71],
[ 73,   5],
```

```
[ 73,  88],
[ 73,   7],
[ 73,  73],
[ 74,  10],
[ 74,  72],
[ 75,   5],
[ 75,  93],
[ 76,  40],
[ 76,  87],
[ 77,  12],
[ 77,  97],
[ 77,  36],
[ 77,  74],
[ 78,  22],
[ 78,  90],
[ 78,  17],
[ 78,  88],
[ 78,  20],
[ 78,  76],
[ 78,  16],
[ 78,  89],
[ 78,   1],
[ 78,  78],
[ 78,   1],
[ 78,  73],
[ 79,  35],
[ 79,  83],
[ 81,   5],
[ 81,  93],
[ 85,  26],
[ 85,  75],
[ 86,  20],
[ 86,  95],
[ 87,  27],
[ 87,  63],
[ 87,  13],
[ 87,  75],
[ 87,  10],
[ 87,  92],
[ 88,  13],
[ 88,  86],
[ 88,  15],
[ 88,  69],
[ 93,  14],
[ 93,  90],
```

```
       [ 97,  32],
       [ 97,  86],
       [ 98,  15],
       [ 98,  88],
       [ 99,  39],
       [ 99,  97],
       [101,  24],
       [101,  68],
       [103,  17],
       [103,  85],
       [103,  23],
       [103,  69],
       [113,   8],
       [113,  91],
       [120,  16],
       [120,  79],
       [126,  28],
       [126,  74],
       [137,  18],
       [137,  83]], dtype=int64)
```

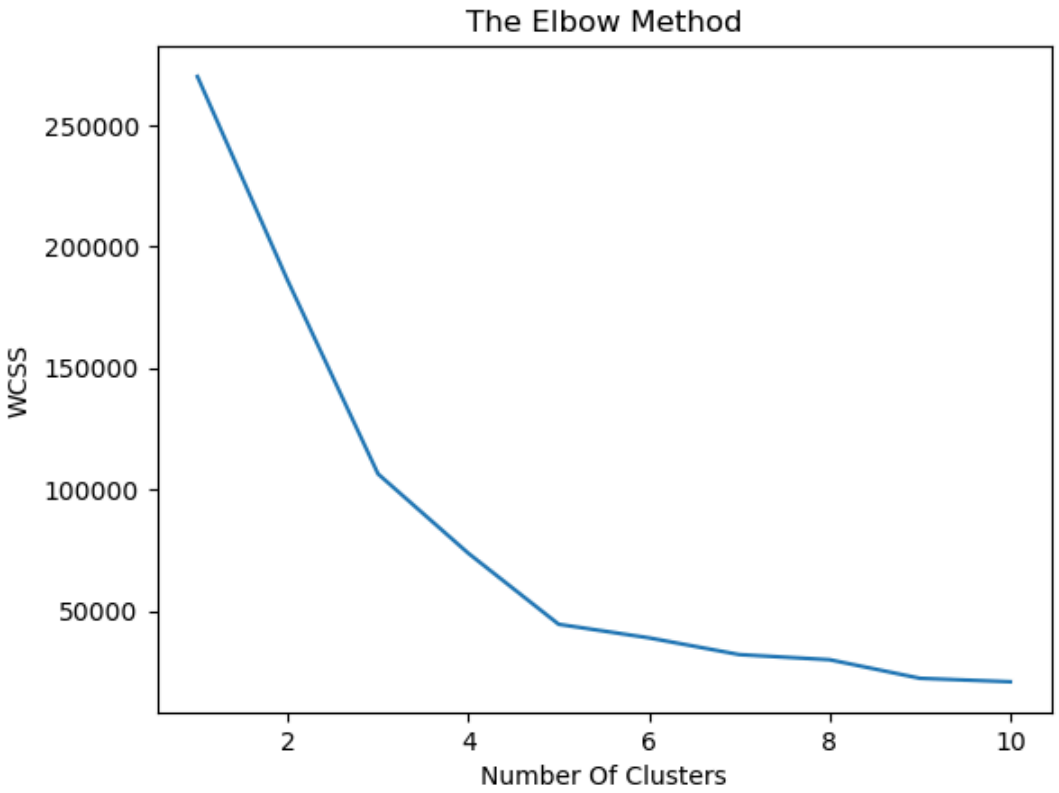
In [6]:
```
# using elbow method to find the optimal number of clusters

from sklearn.cluster import KMeans
```

In [7]:
```
wcss=[]

for i in range(1,11):
    kmeans=KMeans(n_clusters=i,init="k-means++",random_state=0)
    kmeans.fit(x)
    wcss.append(kmeans.inertia_)
plt.plot(range(1,11),wcss)
plt.title('The Elbow Method')
plt.xlabel('Number Of Clusters')
plt.ylabel('WCSS')
plt.show()

# wcss we have very good parameter called inertia_ credit goes to sklearn , that computes the sum of square , formula it will compute
```



In [8]:
```
# training the K-Means model on the dataset

kmeans=KMeans(n_clusters=5,init='k-means++',random_state=0)
kmeans
```

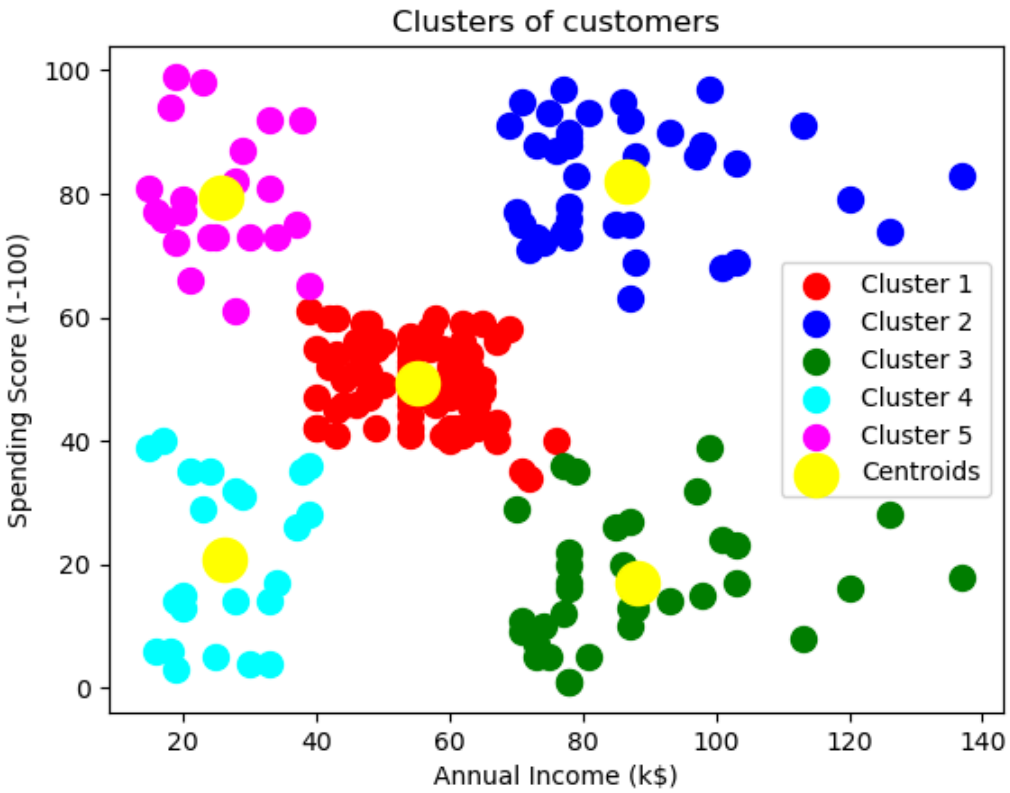
Out[8]:
```
KMeans(n_clusters=5, random_state=0)
```

In [9]:
```
y_kmeans=kmeans.fit_predict(x)
y_kmeans
```

Out[9]:
```
array([3, 4, 3, 4, 3, 4, 3, 4, 3, 4, 3, 4, 3, 4, 3, 4, 3, 4, 3, 4,
       3, 4, 3, 4, 3, 4, 3, 4, 3, 4, 3, 4, 3, 4, 3, 4, 3, 4, 3, 0,
       3, 4, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 2, 1, 0, 1, 2, 1, 2, 1,
       0, 1, 2, 1, 2, 1, 2, 1, 2, 1, 0, 1, 2, 1, 2, 1, 2, 1, 2, 1, 2, 1,
       2, 1, 2, 1, 2, 1, 2, 1, 2, 1, 2, 1, 2, 1, 2, 1, 2, 1, 2, 1, 2, 1,
       2, 1, 2, 1, 2, 1, 2, 1, 2, 1, 2, 1, 2, 1, 2, 1, 2, 1, 2, 1, 2, 1,
       2, 1])
```

In [10]:
```
# visualising the clusters

plt.scatter(x[y_kmeans == 0, 0], x[y_kmeans == 0, 1], s = 100, c = 'red', label = 'Cluster 1')
plt.scatter(x[y_kmeans == 1, 0], x[y_kmeans == 1, 1], s = 100, c = 'blue', label = 'Cluster 2')
plt.scatter(x[y_kmeans == 2, 0], x[y_kmeans == 2, 1], s = 100, c = 'green', label = 'Cluster 3')
plt.scatter(x[y_kmeans == 3, 0], x[y_kmeans == 3, 1], s = 100, c = 'cyan', label = 'Cluster 4')
plt.scatter(x[y_kmeans == 4, 0], x[y_kmeans == 4, 1], s = 100, c = 'magenta', label = 'Cluster 5')
plt.scatter(kmeans.cluster_centers_[:, 0], kmeans.cluster_centers_[:, 1], s = 300, c = 'yellow', label = 'Centroids')
plt.title('Clusters of customers')
plt.xlabel('Annual Income (k$)')
plt.ylabel('Spending Score (1-100)')
plt.legend()
plt.show()
```



In [11]:
```
y_kmeans=kmeans.labels_
dataset['cluster']=y_kmeans
```

In [12]:
```
y_kmeans
```

Out[12]:
```
array([3, 4, 3, 4, 3, 4, 3, 4, 3, 4, 3, 4, 3, 4, 3, 4, 3, 4, 3, 4,
       3, 4, 3, 4, 3, 4, 3, 4, 3, 4, 3, 4, 3, 4, 3, 4, 3, 4, 3, 4, 3, 0,
       3, 4, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 2, 1, 0, 1, 2, 1, 2, 1,
       0, 1, 2, 1, 2, 1, 2, 1, 2, 1, 0, 1, 2, 1, 2, 1, 2, 1, 2, 1, 2, 1,
       2, 1, 2, 1, 2, 1, 2, 1, 2, 1, 2, 1, 2, 1, 2, 1, 2, 1, 2, 1, 2, 1,
       2, 1, 2, 1, 2, 1, 2, 1, 2, 1, 2, 1, 2, 1, 2, 1, 2, 1, 2, 1, 2, 1,
       2, 1])
```

In [13]:
```
# pickling the file

import pickle
filename='Mall_prediction.pkl'
with open (filename,'wb') as file:
    pickle.dump(kmeans,file)
print('Model has been saved and Pickled as Mall_prediction.pkl')
```

```
Model has been saved and Pickled as Mall_prediction.pkl
```

In [14]:
```
import os
os.getcwd()
```

Out[14]:
```
'C:\\Users\\Jan Saida'
```

Streamlit\Mall_Customer_Segmentation_app.py

```python
import streamlit as st
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from sklearn.cluster import KMeans
import pickle

# Load pre-trained KMeans model

filename = 'C:\\Users\Jan Saida\Mall_prediction.pkl'
with open(filename, 'rb') as file:
    kmeans = pickle.load(file)

# Function to load dataset and show initial preview
def load_data():
    dataset = pd.read_csv(r"C:\Users\Jan Saida\OneDrive\Documents\Desktop\Excel sheets\Mall_Customers.csv")
    return dataset

# Title for Streamlit app
st.title("Mall Customer Segmentation Using K-Means Clustering")

# Dataset display
st.header("Dataset Preview")
dataset = load_data()
st.dataframe(dataset.head())

# Allow user to input custom values for clustering
st.sidebar.header("Input Parameters for Clustering")

# Taking input for annual income and spending score
annual_income = st.sidebar.slider("Annual Income (k$)", float(dataset['Annual Income (k$)'].min()), float(dataset['Annual Income (k$)'].max()))
spending_score = st.sidebar.slider("Spending Score (1-100)", 1, 100)

# Display the user's input
st.write(f"User Input - Annual Income: {annual_income}k$ | Spending Score: {spending_score}")

# Predict the cluster based on the user input
user_input = np.array([[annual_income, spending_score]])
cluster_prediction = kmeans.predict(user_input)
```

```
40
41 st.write(f"The customer is predicted to belong to Cluster {cluster_prediction[0] + 1}")
42
43 # Visualize the clusters with the user's input
44 st.header("Cluster Visualization")
45
46 # Get the coordinates of the clusters
47 x = dataset.iloc[:, [3, 4]].values
48 y_kmeans = kmeans.predict(x)
49
50 # Plotting the clusters
51 plt.figure(figsize=(10, 6))
52 plt.scatter(x[y_kmeans == 0, 0], x[y_kmeans == 0, 1], s = 100, c = 'red', label = 'Cluster 1')
53 plt.scatter(x[y_kmeans == 1, 0], x[y_kmeans == 1, 1], s = 100, c = 'blue', label = 'Cluster 2')
54 plt.scatter(x[y_kmeans == 2, 0], x[y_kmeans == 2, 1], s = 100, c = 'green', label = 'Cluster 3')
55 plt.scatter(x[y_kmeans == 3, 0], x[y_kmeans == 3, 1], s = 100, c = 'cyan', label = 'Cluster 4')
56 plt.scatter(x[y_kmeans == 4, 0], x[y_kmeans == 4, 1], s = 100, c = 'magenta', label = 'Cluster 5')
57
58 # Plotting the centroids
59 plt.scatter(kmeans.cluster_centers_[:, 0], kmeans.cluster_centers_[:, 1], s = 300, c = 'yellow', label = 'Centroids')
60
61 # Labels and title
62 plt.title('Clusters of customers')
63 plt.xlabel('Annual Income (k$)')
64 plt.ylabel('Spending Score (1-100)')
65 plt.legend()
66 st.pyplot(plt)
67
68 # Running the app
69 if __name__ == '__main__':
70     st.write("This is a simple app that predicts which customer cluster a person falls into based on their annual income and spending score.")
71
```

Input Parameters for Clustering

Annual Income (k$)

15.00

15.00 137.00

Spending Score (1-100)

39

1 100

Mall Customer Segmentation Using K-Means Clustering

Dataset Preview

	CustomerID	Genre	Age	Annual Income (k$)	Spending Score (1-100)
0	1	Male	19	15	39
1	2	Male	21	15	81
2	3	Female	20	16	6
3	4	Female	23	16	77
4	5	Female	31	17	40

User Input - Annual Income: 15.0k$ | Spending Score: 39

The customer is predicted to belong to Cluster 4

Cluster Visualization

This is a simple app that predicts which customer cluster a person falls into based on their annual income and spending score.

Input Parameters for Clustering

Annual Income (k$)

27.71

15.00 137.00

Spending Score (1-100)

55

1 100

Mall Customer Segmentation Using K-Means Clustering

Dataset Preview

	CustomerID	Genre	Age	Annual Income (k$)	Spending Score (1-100)
0	1	Male	19	15	39
1	2	Male	21	15	81
2	3	Female	20	16	6
3	4	Female	23	16	77
4	5	Female	31	17	40

User Input - Annual Income: 27.71k$ | Spending Score: 55

The customer is predicted to belong to Cluster 5

Cluster Visualization

This is a simple app that predicts which customer cluster a person falls into based on their annual income and spending score.

Input Parameters for Clustering

Annual Income (k$)

58.63

15.00 137.00

Spending Score (1-100)

17

1 100

Mall Customer Segmentation Using K-Means Clustering

Dataset Preview

	CustomerID	Genre	Age	Annual Income (k$)	Spending Score (1-100)
0	1	Male	19	15	39
1	2	Male	21	15	81
2	3	Female	20	16	6
3	4	Female	23	16	77
4	5	Female	31	17	40

User Input - Annual Income: 58.63k$ | Spending Score: 17

The customer is predicted to belong to Cluster 3

Cluster Visualization

This is a simple app that predicts which customer cluster a person falls into based on their annual income and spending score.

Input Parameters for Clustering

Annual Income (k$)

69.22

15.00 137.00

Spending Score (1-100)

33

1 100

Mall Customer Segmentation Using K-Means Clustering

Dataset Preview

	CustomerID	Genre	Age	Annual Income (k$)	Spending Score (1-100)
0	1	Male	19	15	39
1	2	Male	21	15	81
2	3	Female	20	16	6
3	4	Female	23	16	77
4	5	Female	31	17	40

User Input - Annual Income: 69.22k$ | Spending Score: 33

The customer is predicted to belong to Cluster 1

Cluster Visualization

This is a simple app that predicts which customer cluster a person falls into based on their annual income and spending score.

CustomerID	Gender	Age	Annual Income	Spending Score	kmeans Prediction
1	Male	19	15	39	4
2	Male	21	15	81	3
3	Female	20	16	6	4
4	Female	23	16	77	3
5	Female	31	17	40	4
6	Female	22	17	76	3
7	Female	35	18	6	4
8	Female	23	18	94	3
9	Male	64	19	3	4
10	Female	30	19	72	3
11	Male	67	19	14	4
12	Female	35	19	99	3
13	Female	58	20	15	4
14	Female	24	20	77	3
15	Male	37	20	13	4
16	Male	22	20	79	3
17	Female	35	21	35	4
18	Male	20	21	66	3
19	Male	52	23	29	4
20	Female	35	23	98	3
21	Male	35	24	35	4
22	Male	25	24	73	3
23	Female	46	25	5	4
24	Male	31	25	73	3
25	Female	54	28	14	4
26	Male	29	28	82	3
27	Female	45	28	32	4
28	Male	35	28	61	3
29	Female	40	29	31	4
30	Female	23	29	87	3
31	Male	60	30	4	4
32	Female	21	30	73	3
33	Male	53	33	4	4
34	Male	18	33	92	3
35	Female	49	33	14	4
36	Female	21	33	81	3
37	Female	42	34	17	4
38	Female	30	34	73	3
39	Female	36	37	26	4
40	Female	20	37	75	3
41	Female	65	38	35	4
42	Male	24	38	92	3
43	Male	48	39	36	4
44	Female	31	39	61	1
45	Female	49	39	28	4
46	Female	24	39	65	1

47	Female	50	40	55	1
48	Female	27	40	47	1
49	Female	29	40	42	1
50	Female	31	40	42	1
51	Female	49	42	52	1
52	Male	33	42	60	1
53	Female	31	43	54	1
54	Male	59	43	60	1
55	Female	50	43	45	1
56	Male	47	43	41	1
57	Female	51	44	50	1
58	Male	69	44	46	1
59	Female	27	46	51	1
60	Male	53	46	46	1
61	Male	70	46	56	1
62	Male	19	46	55	1
63	Female	67	47	52	1
64	Female	54	47	59	1
65	Male	63	48	51	1
66	Male	18	48	59	1
67	Female	43	48	50	1
68	Female	68	48	48	1
69	Male	19	48	59	1
70	Female	32	48	47	1
71	Male	70	49	55	1
72	Female	47	49	42	1
73	Female	60	50	49	1
74	Female	60	50	56	1
75	Male	59	54	47	1
76	Male	26	54	54	1
77	Female	45	54	53	1
78	Male	40	54	48	1
79	Female	23	54	52	1
80	Female	49	54	42	1
81	Male	57	54	51	1
82	Male	38	54	55	1
83	Male	67	54	41	1
84	Female	46	54	44	1
85	Female	21	54	57	1
86	Male	48	54	46	1
87	Female	55	57	58	1
88	Female	22	57	55	1
89	Female	34	58	60	1
90	Female	50	58	46	1
91	Female	68	59	55	1
92	Male	18	59	41	1
93	Male	48	60	49	1

94	Female	40	60	40	1
95	Female	32	60	42	1
96	Male	24	60	52	1
97	Female	47	60	47	1
98	Female	27	60	50	1
99	Male	48	61	42	1
100	Male	20	61	49	1
101	Female	23	62	41	1
102	Female	49	62	48	1
103	Male	67	62	59	1
104	Male	26	62	55	1
105	Male	49	62	56	1
106	Female	21	62	42	1
107	Female	66	63	50	1
108	Male	54	63	46	1
109	Male	68	63	43	1
110	Male	66	63	48	1
111	Male	65	63	52	1
112	Female	19	63	54	1
113	Female	38	64	42	1
114	Male	19	64	46	1
115	Female	18	65	48	1
116	Female	19	65	50	1
117	Female	63	65	43	1
118	Female	49	65	59	1
119	Female	51	67	43	1
120	Female	50	67	57	1
121	Male	27	67	56	1
122	Female	38	67	40	1
123	Female	40	69	58	1
124	Male	39	69	91	2
125	Female	23	70	29	1
126	Female	31	70	77	2
127	Male	43	71	35	1
128	Male	40	71	95	2
129	Male	59	71	11	0
130	Male	38	71	75	2
131	Male	47	71	9	0
132	Male	39	71	75	2
133	Female	25	72	34	1
134	Female	31	72	71	2
135	Male	20	73	5	0
136	Female	29	73	88	2
137	Female	44	73	7	0
138	Male	32	73	73	2
139	Male	19	74	10	0
140	Female	35	74	72	2

141	Female	57	75	5	0
142	Male	32	75	93	2
143	Female	28	76	40	1
144	Female	32	76	87	2
145	Male	25	77	12	0
146	Male	28	77	97	2
147	Male	48	77	36	1
148	Female	32	77	74	2
149	Female	34	78	22	0
150	Male	34	78	90	2
151	Male	43	78	17	0
152	Male	39	78	88	2
153	Female	44	78	20	0
154	Female	38	78	76	2
155	Female	47	78	16	0
156	Female	27	78	89	2
157	Male	37	78	1	0
158	Female	30	78	78	2
159	Male	34	78	1	0
160	Female	30	78	73	2
161	Female	56	79	35	1
162	Female	29	79	83	2
163	Male	19	81	5	0
164	Female	31	81	93	2
165	Male	50	85	26	0
166	Female	36	85	75	2
167	Male	42	86	20	0
168	Female	33	86	95	2
169	Female	36	87	27	0
170	Male	32	87	63	2
171	Male	40	87	13	0
172	Male	28	87	75	2
173	Male	36	87	10	0
174	Male	36	87	92	2
175	Female	52	88	13	0
176	Female	30	88	86	2
177	Male	58	88	15	0
178	Male	27	88	69	2
179	Male	59	93	14	0
180	Male	35	93	90	2
181	Female	37	97	32	0
182	Female	32	97	86	2
183	Male	46	98	15	0
184	Female	29	98	88	2
185	Female	41	99	39	0
186	Male	30	99	97	2
187	Female	54	101	24	0

188	Male	28	101	68	2
189	Female	41	103	17	0
190	Female	36	103	85	2
191	Female	34	103	23	0
192	Female	32	103	69	2
193	Male	33	113	8	0
194	Female	38	113	91	2
195	Female	47	120	16	0
196	Female	35	120	79	2
197	Female	45	126	28	0
198	Male	32	126	74	2
199	Male	32	137	18	0
200	Male	30	137	83	2