1. Install Python and Others Packages

- Install Python check: `python --version` and `py --version`
- Install Virtualenv: `pip install virtualenv` and `virtualenv venv`

In [48]:
```
# !pip install numpy pandas matplotlib seaborn scikit-learn scipy statsmodels jupyt
```

- Run Jupyterlab like: `jupyter lab`

2. Import Necessary Packages for Python

In [49]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns

from scipy import stats
from scipy.stats import pearsonr, shapiro

import statsmodels.api as sm
from sklearn.linear_model import LinearRegression, LogisticRegression
```

3. Dataset

In [50]:
```
data = {
    "Subject": range(1, 15),
    "GPA": [3.8, 4.0, 3.2, 3.5, 2.5, 3.0, 2.1, 2.8, 3.6, 4.0, 3.6, 3.4, 3.2, 2.0],
    "Adaptability": [45, 50, 45, 51, 60, 39, 42, 41, 46, 50, 53, 47, 48, 40],
    "Self Confidence": [60, 10, 50, 25, 15, 80, 41, 14, 57, 68, 24, 95, 25, 36],
    "IQ": [105, 109, 102, 95, 92, 101, 99, 95, 94, 110, 104, 105, 98, 75],
    "Gender": ["Female", "Female", "Female", "Female", "Male" , "Male", "Male", "Fe
    "Economic Condition": ["Good", "Good", "Poor", "Good", 'Poor', 'Middle', 'Poor'
}

student_df = pd.DataFrame(data)
student_df.head()
```

Out[50]:

	Subject	GPA	Adaptability	Self Confidence	IQ	Gender	Economic Condition
0	1	3.8	45	60	105	Female	Good
1	2	4.0	50	10	109	Female	Good
2	3	3.2	45	50	102	Female	Poor
3	4	3.5	51	25	95	Female	Good
4	5	2.5	60	15	92	Male	Poor

In [51]:

```
data = {
    "Year": range(2010, 2022),
    "Production (thousands of kg)": [60.04, 59.13, 62.52, 66.26, 63.88, 67.38, 85.9
    "Consumption (in thousands of kg)": [57.63, 58.50, 61.19, 64.00, 67.17, 77.57,
}

production_df = pd.DataFrame(data)
production_df.head()
```

Out[51]:

	Year	Production (thousands of kg)	Consumption (in thousands of kg)
0	2010	60.04	57.63
1	2011	59.13	58.50
2	2012	62.52	61.19
3	2013	66.26	64.00
4	2014	63.88	67.17

In [52]:

```
data = {
    "ID": [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20],
    "Age": [48, 50, 46, 44, 60, 41, 49, 45, 55, 51, 65, 54, 44, 42, 53, 44, 44, 50,
    "Height (in inch)": [61, 62, 60, 57, 58, 55, 56, 62, 60, 67, 54, 72, 69, 68, 59
    "IQ": [110, 100, 102, 92, 95, 108, 92, 110, 93, 115, 115, 130, 122, 125, 130, 1
    "BMI": [24.99, 20.16, 22.39, 20.04, 20.73, 29.72, 20.76, 24.19, 18.51, 22.44, 3
    "Gender": ["Male", "Male", "Male", "Female", "Female", "Female", "Female", "Fem
    "Family Type": ["Nuclear", "Nuclear", "Nuclear", "Nuclear", "Joint", "Joint", "
    "Smoking Habit": ["Yes", "Yes", "Yes", "No", "No", "No", "No", "No", "No", "No"
    "Disease Suffering": ["Yes", "Yes", "No", "Yes", "No", "No", "No", "No", "Yes",
    "Psychological Stress": ["Yes", "Yes", "Yes", "Yes", "Yes", "No", "Yes", "No",
}

person_df = pd.DataFrame(data)
person_df.head()
```

Out[52]:

	ID	Age	Height (in inch)	IQ	BMI	Gender	Family Type	Smoking Habit	Disease Suffering	Psychological Stress
0	1	48	61	110	24.99	Male	Nuclear	Yes	Yes	Yes
1	2	50	62	100	20.16	Male	Nuclear	Yes	Yes	Yes
2	3	46	60	102	22.39	Male	Nuclear	Yes	No	Yes
3	4	44	57	92	20.04	Female	Nuclear	No	Yes	Yes
4	5	60	58	95	20.73	Female	Joint	No	No	Yes

4. Solve Suggestion

1. Normality test for Continuous data, then regression.
2. Linear regression for GPA, self confidence and IQ
3. Logistic regression for Self confidence, Gender and Economic Condition.
4. Cronbach's alpha test for categorical data

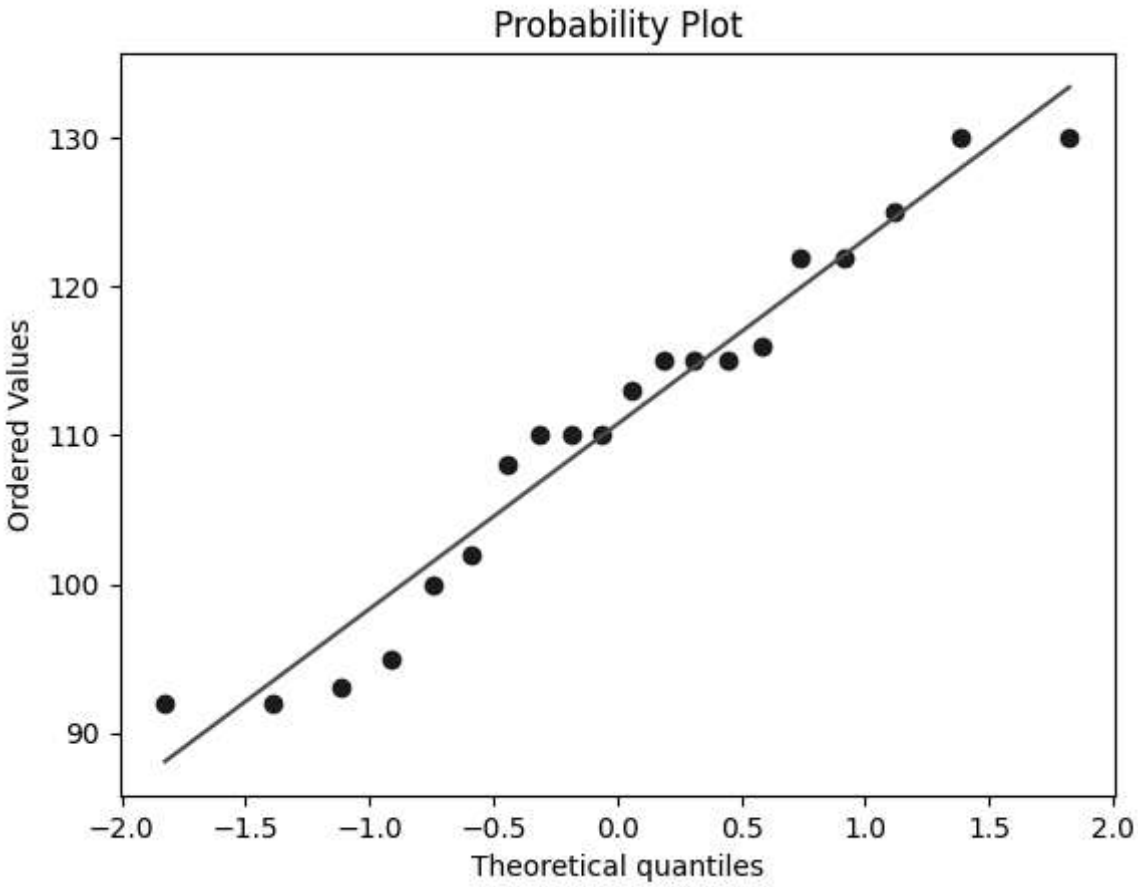
4.1. Normality test for Continuous Data, then regression

Null Hypothesis (H_0): The data follows a normal distribution.

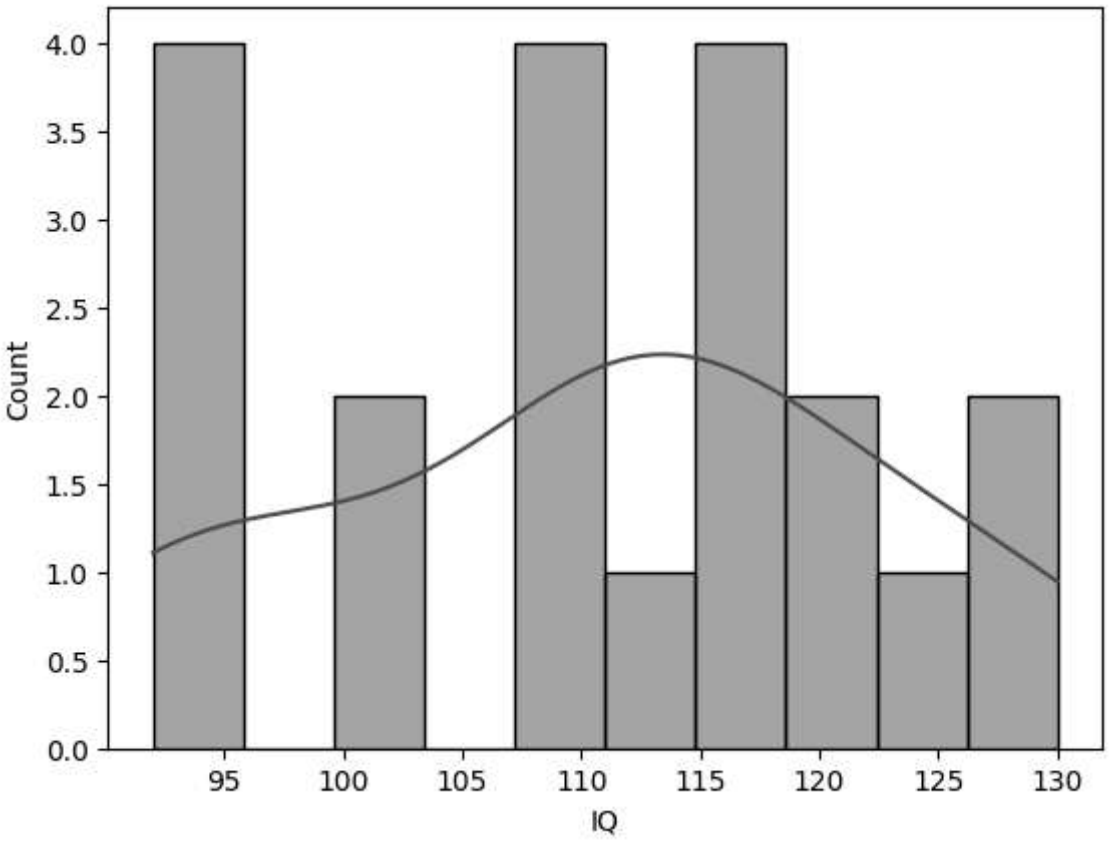
Alternative Hypothesis (H_1): The data does not follow a normal distribution.

In [53]:
```
# Normality test for person_df['IQ']
from scipy.stats import shapiro

data = person_df['IQ']

# Q-Q Plot
stats.probplot(data, dist="norm", plot= plt)
plt.show()

#Shapiro-Wilk Test
shapiro_value, p_value = shapiro(data)

print("Shapiro value is:", shapiro_value)
print("P-Value is:", np.round(p_value, 4))
if p_value <= 0.05:
    print("Reject the null hypothesis at the 5% significance level.")
    print("Data does not follow normal distribution.")
else:
    print("Can't Reject the null hypothesis at the 5% significance level.")
    print("Data follow a normal distribution")
```

```
# Histogram
sns.histplot(data, kde= True, bins=10, color="red", edgecolor="black")
plt.show()
```




```
Shapiro value is: 0.9422618973900575
P-Value is: 0.2645
Can't Reject the null hypothesis at the 5% significance level.
Data follow a normal distribution
```



- Shapiro-Wilk Test Statistic: 0.9423
- P-Value: 0.2645

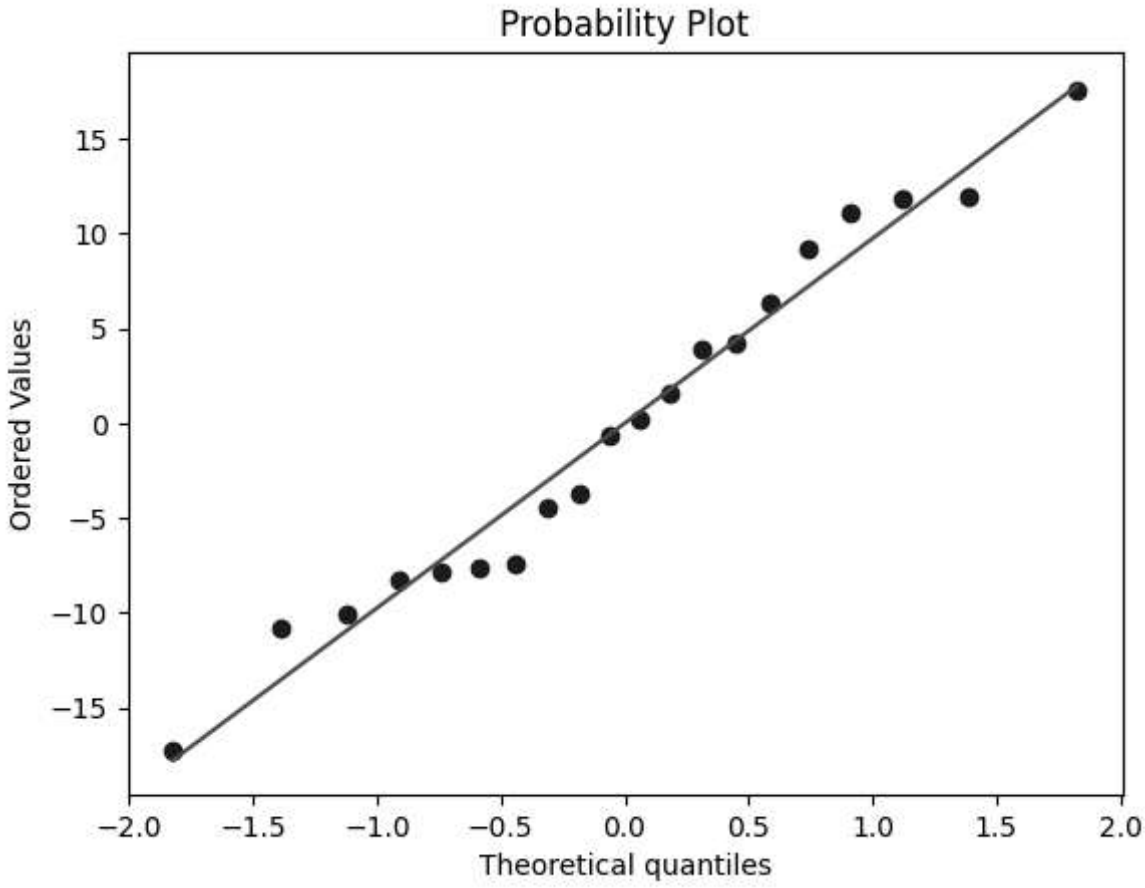
Since the p-value is greater than the common significance level of 0.05, we fail to reject the null hypothesis. This means there is no strong evidence to suggest that the IQ data is not normally distributed. Therefore, we can assume that the IQ data follows a normal distribution.

4.1.1. Check normality using Regression

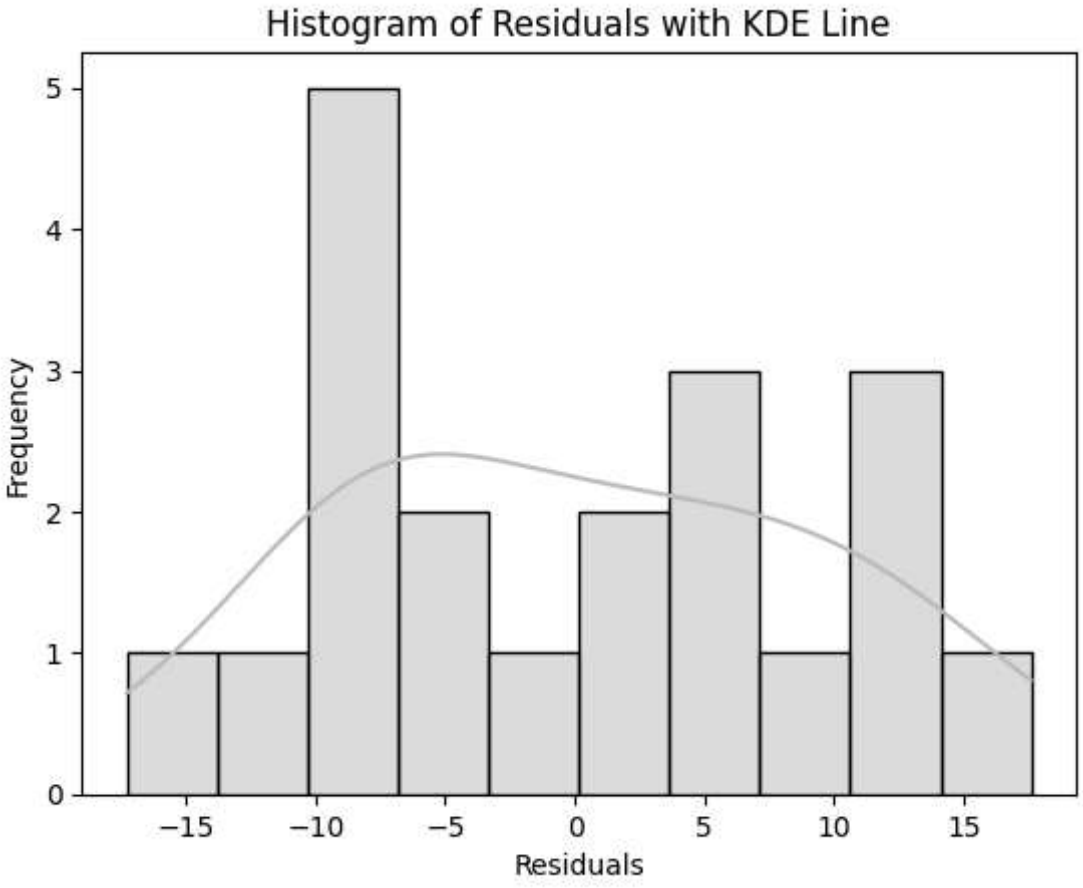
In [54]:
```
import statsmodels.api as sm

y = person_df['IQ']
X = person_df[['Age', 'Height (in inch)', 'BMI']]
X = sm.add_constant(X)

model = sm.OLS(y, X).fit()
residuals = model.resid

print(model.summary())
```

```
                            OLS Regression Results
==============================================================================
Dep. Variable:                     IQ   R-squared:                       0.393
Model:                            OLS   Adj. R-squared:                  0.279
Method:                 Least Squares   F-statistic:                     3.457
Date:                Wed, 09 Apr 2025   Prob (F-statistic):             0.0415
Time:                        16:33:32   Log-Likelihood:                -72.625
No. Observations:                  20   AIC:                             153.2
Df Residuals:                      16   BIC:                             157.2
Df Model:                           3
Covariance Type:            nonrobust
==========================================================================================
                             coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------------------
const                    54.8970     28.136      1.951      0.069      -4.749     114.543
Age                      -0.0107      0.385     -0.028      0.978      -0.826       0.805
Height (in inch)          0.3128      0.206      1.520      0.148      -0.124       0.749
BMI                       1.4871      0.514      2.891      0.011       0.397       2.578
==============================================================================
Omnibus:                        0.979   Durbin-Watson:                   1.639
Prob(Omnibus):                  0.613   Jarque-Bera (JB):                0.774
Skew:                           0.126   Prob(JB):                        0.679
Kurtosis:                       2.070   Cond. No.                     1.05e+03
==============================================================================

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly spe
cified.
[2] The condition number is large, 1.05e+03. This might indicate that there are
strong multicollinearity or other numerical problems.
```

In [55]:
```python
# QQ-Plot
stats.probplot(residuals, dist="norm", plot=plt)
plt.show()

# Shapiro-Wilk Test
shapiro_value, p_value = shapiro(data)
print("Shapiro value is:", shapiro_value)
print("P-Value is:", np.round(p_value, 4))
if p_value <= 0.05:
    print("Reject the null hypothesis at the 5% significance level.")
    print("Data does not follow normal distribution.")
else:
    print("Can't Reject the null hypothesis at the 5% significance level.")
    print("Data follow a normal distribution")

# Histogram with KDE line
sns.histplot(residuals, kde=True, bins=10, color='skyblue', edgecolor='black')
plt.title("Histogram of Residuals with KDE Line")
plt.xlabel("Residuals")
plt.ylabel("Frequency")
plt.show()
```



Shapiro value is: 0.9422618973900575
P-Value is: 0.2645
Can't Reject the null hypothesis at the 5% significance level.
Data follow a normal distribution



4.2. Linear regression for GPA, self confidence and IQ

In [56]:
```
# Linear Regression for GPA with Scikit-Learn
from sklearn.linear_model import LinearRegression

y = student_df['GPA']
X = student_df[['Adaptability', 'Self Confidence', 'IQ']]

model = LinearRegression()
model.fit(X, y)

intercept = model.intercept_
coefficient = model.coef_
print(f"Linear Regression Model Equation: {intercept:.2f} + ({coefficient[0]:.2f} *
```

```
Linear Regression Model Equation: -2.85 + (0.02 * Adaptability)+ (0.00 * Self Confid
ence)+ (0.05 * IQ)
```

In [57]:
```
# Linear Regression for GPA with Statsmodels
import statsmodels.api as sm

y = student_df['GPA']
X = student_df[['Adaptability', 'Self Confidence', 'IQ']]
X = sm.add_constant(X)
```

```
model = sm.OLS(y, X).fit()
print(model.summary())
```

```
                            OLS Regression Results
==============================================================================
Dep. Variable:                    GPA   R-squared:                       0.584
Model:                            OLS   Adj. R-squared:                  0.460
Method:                 Least Squares   F-statistic:                     4.685
Date:                Wed, 09 Apr 2025   Prob (F-statistic):             0.0271
Time:                        16:33:32   Log-Likelihood:                -7.0902
No. Observations:                  14   AIC:                             22.18
Df Residuals:                      10   BIC:                             24.74
Df Model:                           3
Covariance Type:            nonrobust
================================================================================
                     coef    std err          t      P>|t|      [0.025      0.975]
--------------------------------------------------------------------------------
const             -2.8511      1.677     -1.700      0.120      -6.588       0.885
Adaptability       0.0220      0.027      0.824      0.429      -0.038       0.082
Self Confidence    0.0016      0.006      0.265      0.796      -0.012       0.015
IQ                 0.0500      0.017      2.984      0.014       0.013       0.087
==============================================================================
Omnibus:                        4.306   Durbin-Watson:                   1.827
Prob(Omnibus):                  0.116   Jarque-Bera (JB):                1.908
Skew:                          -0.851   Prob(JB):                        0.385
Kurtosis:                       3.610   Cond. No.                     1.56e+03
==============================================================================

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly spe
cified.
[2] The condition number is large, 1.56e+03. This might indicate that there are
strong multicollinearity or other numerical problems.
```

```
F:\STAT42\ResearchMethodology\lab2_python\venv\Lib\site-packages\scipy\stats\_axis_n
an_policy.py:430: UserWarning: `kurtosistest` p-value may be inaccurate with fewer t
han 20 observations; only n=14 observations were given.
  return hypotest_fun_in(*args, **kwds)
```

Key Model Metrics:

1. **Dependent Variable (Dep. Variable)**: GPA is the variable being predicted based on the given independent variables (Adaptability, Self Confidence, IQ).
2. **R-squared**: The model explains 58.4% of the variability in GPA, meaning these predictors moderately contribute to GPA variance. The **Adjusted R-squared** (46.0%) accounts for the number of predictors, showing a slightly lower explanatory power when adjusted for overfitting.
3. **F-statistic (4.685)**: The F-test evaluates if the model provides better predictions than a model with no predictors. The **Prob (F-statistic)** value of 0.0271 indicates the overall model is statistically significant at a 5% significance level.
4. **AIC/BIC**: These are metrics assessing model fit, penalizing for complexity. Lower values suggest better model performance relative to other potential models.

Coefficients Analysis:

1. **Intercept (const)**:
    - Value: -2.8511. This represents the baseline GPA when all predictors (Adaptability, Self Confidence, IQ) are zero.
    - P-value (0.120): Indicates the intercept is not statistically significant (greater than 0.05). It's likely that the baseline GPA should not be interpreted without proper context.
    - Confidence Interval (-6.588, 0.885): A wide range shows uncertainty about the baseline.
2. **Adaptability**:
    - Coefficient: 0.0220. Suggests a unit increase in Adaptability leads to a predicted 0.022 increase in GPA, all else constant.
    - P-value (0.429): Shows that Adaptability is not statistically significant as a predictor.
    - Confidence Interval (-0.038, 0.082): Includes zero, reaffirming its insignificance.
3. **Self Confidence**:
    - Coefficient: 0.0016. This reflects an almost negligible effect of Self Confidence on GPA.
    - P-value (0.796): Indicates Self Confidence is also statistically insignificant.
    - Confidence Interval (-0.012, 0.015): Includes zero, reaffirming no meaningful contribution.
4. **IQ**:
    - Coefficient: 0.0500. Suggests each unit increase in IQ leads to a predicted 0.05 rise in GPA, all else constant.
    - P-value (0.014): Indicates IQ is a statistically significant predictor at the 5% level.
    - Confidence Interval (0.013, 0.087): Confirms that the positive effect of IQ on GPA is consistent.

Residual Analysis:

- **Durbin-Watson Statistic (1.827)**: Tests for autocorrelation in residuals. A value near 2 indicates no autocorrelation, which is favorable.
- **Omnibus and Jarque-Bera tests**: Assess normality of residuals. The p-values (>0.05) suggest residuals may follow a normal distribution, which supports the validity of inference.

Interpretation of Results:

- The model moderately explains GPA variability (R-squared: 58.4%).
- Among the predictors, **IQ** is the only statistically significant factor influencing GPA, with a positive effect.
- Both **Adaptability** and **Self Confidence** lack significant contributions to predicting GPA in this dataset.

- The results suggest IQ plays a key role in GPA determination, but the other factors may require further investigation with larger sample sizes or alternative methods.

4.3. Logistic regression for Self confidence, Gender and Economic Condition.

In [58]:
```
#Student Dataset
student_df.head()
student_df_encoded = student_df.copy()
student_df_encoded['Gender'] = student_df_encoded['Gender'].apply(lambda x: 1 if x
student_df_encoded['Economic Condition'] = student_df_encoded['Economic Condition']
student_df_encoded.head()
```

Out[58]:

	Subject	GPA	Adaptability	Self Confidence	IQ	Gender	Economic Condition
0	1	3.8	45	60	105	0	3
1	2	4.0	50	10	109	0	3
2	3	3.2	45	50	102	0	1
3	4	3.5	51	25	95	0	3
4	5	2.5	60	15	92	1	1

In [59]:
```
# Logistic Regresion for Gender using Scikit-learn
from sklearn.linear_model import LogisticRegression

y = student_df_encoded['Gender']
X = student_df_encoded[['GPA', 'Adaptability', 'IQ']]

model = LogisticRegression()
model.fit(X, y)


# Coefficients of the logistic regression model
coefficients = model.coef_
intercept = model.intercept_
print(f"Logistic Regression Equation = {intercept[0]:.2f} + ({coefficients[0][0]:.2
```

```
Logistic Regression Equation = 17.32 + (-1.04 * GPA) + (-0.02 * Adaptability) + (-0.
14 * IQ)
```

In [60]:
```
# Logistic Regression for Gender using statsmodels
import statsmodels.api as sm

y = student_df_encoded['Gender']
X = student_df_encoded[['GPA', 'Adaptability', 'IQ']]
X = sm.add_constant(X)

model = sm.Logit(y, X).fit()
print(model.summary())
```

```
Optimization terminated successfully.
         Current function value: 0.280298
         Iterations 9
                           Logit Regression Results
==============================================================================
Dep. Variable:                 Gender   No. Observations:                   14
Model:                          Logit   Df Residuals:                       10
Method:                           MLE   Df Model:                            3
Date:                Wed, 09 Apr 2025   Pseudo R-squ.:                  0.5699
Time:                        16:33:32   Log-Likelihood:                -3.9242
converged:                       True   LL-Null:                       -9.1246
Covariance Type:            nonrobust   LLR p-value:                   0.01545
==============================================================================
                 coef    std err          z      P>|z|      [0.025      0.975]
------------------------------------------------------------------------------
const          4.1325     23.735      0.174      0.862     -42.388      50.653
GPA           -6.7259      6.274     -1.072      0.284     -19.023       5.572
Adaptability   0.1131      0.300      0.377      0.706      -0.475       0.701
IQ             0.1127      0.310      0.364      0.716      -0.495       0.720
==============================================================================
```

The output you've provided is the result of a logistic regression model fitted using the `statsmodels` library. The dependent variable is `Gender`, and the independent variables are `GPA`, `Adaptability`, and `IQ`. Here is a detailed interpretation of the key statistics:

1. **Model Fit Information**

- **Dep. Variable: Gender**: The dependent variable (target) is `Gender`, indicating that the goal of the model is to predict the probability of an individual being of a certain gender (likely binary: male/female).
- **No. Observations: 14**: The dataset has 14 observations, which is quite small. In general, for statistical models, having a larger sample size is preferable to ensure the model's robustness.
- **Df Residuals: 10**: The degrees of freedom for the residuals is 10, which is derived from the total number of observations (14) minus the number of parameters estimated in the model (4, including the intercept).
- **Df Model: 3**: This represents the number of independent variables in the model (GPA, Adaptability, IQ).
- **Log-Likelihood: -3.9242**: This is the log-likelihood of the model, which tells us how well the model fits the data. Higher values of the log-likelihood indicate a better fit.
- **Pseudo R-squ.: 0.5699**: The pseudo R-squared value indicates the proportion of variance in the dependent variable that is explained by the model. A pseudo R-squared of 0.5699 suggests that the model explains about 56.99% of the variance in `Gender`. This is considered a moderate fit for logistic regression, but generally, pseudo R-squared values do not have the same interpretation as R-squared in linear regression.
- **LLR p-value: 0.01545**: The likelihood ratio test (LLR) p-value tests the null hypothesis that the model with no predictors fits the data as well as the current model. Since the p-value is 0.015, which is less than 0.05, this suggests that the model with the predictors

(`GPA`, `Adaptability`, `IQ`) significantly improves the fit compared to a model with no predictors.

2. **Coefficients and Interpretation**

The coefficients represent the log-odds of the dependent variable being 1 (let's assume `Gender = 1` represents one gender, e.g., male). Here's a breakdown of the key results:

- **Intercept (const): 4.1325**
  - The intercept represents the log-odds of the outcome when all the independent variables (GPA, Adaptability, IQ) are zero. In this case, the log-odds of being in one gender when GPA, Adaptability, and IQ are all zero is 4.1325.
  - However, this intercept is large and has a high standard error (23.735), leading to a high p-value of 0.862. This means that the intercept is not statistically significant, and the estimate of the intercept itself isn't reliable. This suggests that the baseline gender outcome isn't clearly defined when all predictors are zero.
- **GPA: -6.7259**
  - The coefficient for GPA is negative and quite large in magnitude. This suggests that for each one-unit increase in GPA, the log-odds of being in one gender decreases by 6.7259 units, which is a very steep decrease. However, the p-value for GPA is 0.284, which is much greater than 0.05, meaning that this effect is not statistically significant. The large standard error (6.274) also indicates that the estimate is not precise.
  - Given the lack of statistical significance, we cannot conclude that GPA has a meaningful effect on predicting gender.
- **Adaptability: 0.1131**
  - The coefficient for Adaptability is positive, indicating that higher adaptability is associated with slightly increased log-odds of being in one gender. However, the effect size is small (0.1131), and the p-value is 0.706, which is much greater than 0.05. This suggests that Adaptability does not have a statistically significant effect on the gender prediction.
- **IQ: 0.1127**
  - The coefficient for IQ is also positive, indicating that higher IQ is associated with slightly increased log-odds of being in one gender. However, the magnitude of the coefficient (0.1127) is small, and the p-value (0.716) suggests that this effect is not statistically significant.

3. **Statistical Significance**

- None of the predictors (GPA, Adaptability, IQ) have statistically significant p-values (all are greater than 0.05). This indicates that, at least based on this dataset, none of these variables are strong predictors of gender.

- The standard errors for the coefficients are relatively high, which further weakens the reliability of the estimated coefficients.

4. Conclusion

- **Model Performance**: The model shows some improvement over a null model, as indicated by the likelihood ratio test (LLR p-value = 0.01545). However, with a small sample size (14 observations), the statistical power is limited, and it's difficult to draw strong conclusions.
- **Variable Significance**: None of the predictors (GPA, Adaptability, IQ) are statistically significant in predicting gender in this sample. This means that we cannot confidently say that any of these variables are useful for gender classification in this particular dataset.
- **Next Steps**:
  - A larger dataset would improve the reliability and power of the model.
  - You may want to explore other variables or consider interactions between predictors.
  - Model tuning, such as regularization (e.g., Lasso or Ridge regression), could help to reduce overfitting or address multicollinearity, especially if you have a larger number of predictors in future models.

Comments:

- The small sample size and large standard errors in the coefficients limit the conclusions that can be drawn from this model.
- It is advisable to explore more data, consider the relevance of the predictors, and potentially look into other statistical models or transformations that may better capture the relationship between the predictors and the outcome variable.

In [61]:
```
#Logistic Regression for Economic Condition
y = student_df_encoded['Economic Condition']
X = student_df_encoded[['GPA', 'Adaptability', 'Self Confidence', 'IQ']]

from statsmodels.miscmodels.ordinal_model import OrderedModel
model = OrderedModel(y, X, distr='logit').fit()
print(model.summary())
```

```
                           OrderedModel Results
==============================================================================
Dep. Variable:     Economic Condition   Log-Likelihood:                -5.4414
Model:                   OrderedModel   AIC:                             22.88
Method:            Maximum Likelihood   BIC:                             26.72
Date:                Wed, 09 Apr 2025
Time:                        16:33:32
No. Observations:                  14
Df Residuals:                       8
Df Model:                           4
==============================================================================
                     coef    std err          z      P>|z|      [0.025      0.975]
------------------------------------------------------------------------------
GPA                8.4074      4.371      1.924      0.054      -0.159      16.974
Adaptability       0.0399      0.277      0.144      0.885      -0.502       0.582
Self Confidence   -0.0198      0.043     -0.459      0.646      -0.104       0.065
IQ                 0.1045      0.226      0.463      0.644      -0.338       0.547
1/2               36.7356     22.772      1.613      0.107      -7.897      81.368
2/3                1.4562      0.541      2.689      0.007       0.395       2.518
==============================================================================
```

```
F:\STAT42\ResearchMethodology\lab2_python\venv\Lib\site-packages\statsmodels\base\optimizer.py:737: RuntimeWarning: Maximum number of iterations has been exceeded.
  retvals = optimize.fmin(f, start_params, args=fargs, xtol=xtol,
F:\STAT42\ResearchMethodology\lab2_python\venv\Lib\site-packages\statsmodels\base\model.py:607: ConvergenceWarning: Maximum Likelihood optimization failed to converge. Check mle_retvals
  warnings.warn("Maximum Likelihood optimization failed to "
```

🔢 Key Model Metrics:

- **Log-Likelihood**: -5.4414
- **AIC**: 22.88
- **BIC**: 26.72
- **No. of Observations**: 14 (small sample size — keep in mind for interpretation)
- **Model Type**: Ordered Logistic Regression (with Logit link)

📊 Coefficient Estimates:

Variable	Coefficient	p-value	Interpretation
GPA	8.4074	0.054	Positively associated with better economic condition; borderline significant at 5% level.
Adaptability	0.0399	0.885	Not significant; little to no effect.
Self Confidence	-0.0198	0.646	Not significant; minimal negative association.
IQ	0.1045	0.644	Not significant; weak effect.

- Only **GPA** shows a **marginally significant** positive effect on economic condition ($p \approx 0.054$), suggesting students with higher GPA may be more likely to have a better

economic condition.

- All other variables have **p-values > 0.6**, indicating **no statistically significant effect**.

Comment:

The ordered logistic regression model suggests that **GPA might be a meaningful predictor** of a student's economic condition, but the result is only **marginally significant** (p = 0.054). The other predictors (Adaptability, Self Confidence, and IQ) show **no significant influence** on economic condition in this small sample of 14 observations.

4.4. Cronbach's alpha test for categorical data

Formula: $\alpha = \frac{k}{(k-1)} (1- \frac{\sum \sigma_i^2}{\sigma_T^2})$

Interpreting Cronbach's Alpha:

α ≥ 0.9 → Excellent reliability

0.8 ≤ α < 0.9 → Good reliability

0.7 ≤ α < 0.8 → Acceptable

0.6 ≤ α < 0.7 → Questionable

0.5 ≤ α < 0.6 → Poor

α < 0.5 → Unacceptable

In [62]:
```
# Select numerical variables for analysis
numerical_df = student_df[['GPA', 'Adaptability', 'Self Confidence', 'IQ']]

def cronbach_alpha(df):
    k = df.shape[1]
    item_variance = df.var(axis="rows", ddof=1)
    total_score = df.sum(axis="columns")
    total_variance = total_score.var(ddof = 1)
    alpha = (k / (k-1)) * (1 - (item_variance.sum()/total_variance))
    return alpha

print("Cornbach's Alpha:", cronbach_alpha(numerical_df))
```

```
Cornbach's Alpha: 0.0951452322552558
```

In [63]:
```
student_df_cat = student_df_encoded[['GPA', 'Economic Condition']]

def cronbach_alpha(df):
    k = df.shape[1]
    item_variance = df.var(axis="rows", ddof=1)
    total_score = df.sum(axis="columns")
    total_variance = total_score.var(ddof = 1)
```

```
    alpha = (k / (k-1)) * (1 - (item_variance.sum()/total_variance))
    return alpha

print("Cornbach's Alpha:", cronbach_alpha(student_df_cat))
```

```
Cornbach's Alpha: 0.8990596569914122
```

In [64]:
```
#Using pingouin Library
from pingouin import cronbach_alpha

numerical_df = student_df[['GPA', 'IQ']]
alpha = cronbach_alpha(numerical_df)

print("Cronbach's Alpha:", alpha[0])
```

```
Cronbach's Alpha: 0.19575101432029962
```

A Cronbach's alpha of 0.196 indicates very poor internal consistency between the two variables (GPA and IQ). This means:

- Low Reliability: These two variables do not measure the same underlying construct (e.g., they are not part of a coherent scale).
- Unrelated Constructs: GPA (academic performance) and IQ (cognitive ability) are likely measuring distinct traits, and their responses vary independently.

5. 2022 Question

1.i) Identify Influential Factors of academic performance of students.

In [65]:
```
import statsmodels.api as sm

y = student_df_encoded['GPA']
X = student_df_encoded[['Adaptability', 'Self Confidence', 'IQ', 'Economic Conditio
X = sm.add_constant(X)

model = sm.OLS(y, X).fit()
print(model.summary())
```

```
                            OLS Regression Results
==============================================================================
Dep. Variable:                    GPA   R-squared:                       0.820
Model:                            OLS   Adj. R-squared:                  0.740
Method:                 Least Squares   F-statistic:                     10.27
Date:                Wed, 09 Apr 2025   Prob (F-statistic):            0.00207
Time:                        16:33:32   Log-Likelihood:                -1.2195
No. Observations:                  14   AIC:                             12.44
Df Residuals:                       9   BIC:                             15.63
Df Model:                           4
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
const         -0.6144      1.332     -0.461      0.656      -3.627       2.399
Adaptability   0.0071      0.019      0.375      0.716      -0.036       0.050
Self Confidence  0.0011      0.004      0.280      0.786      -0.008       0.010
IQ             0.0253      0.014      1.856      0.096      -0.006       0.056
Economic Condition  0.4595      0.134      3.438      0.007       0.157       0.762
==============================================================================
Omnibus:                        1.397   Durbin-Watson:                   1.635
Prob(Omnibus):                  0.497   Jarque-Bera (JB):                0.139
Skew:                          -0.088   Prob(JB):                        0.933
Kurtosis:                       3.455   Cond. No.                     1.79e+03
==============================================================================

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
[2] The condition number is large, 1.79e+03. This might indicate that there are
strong multicollinearity or other numerical problems.
```

```
F:\STAT42\ResearchMethodology\lab2_python\venv\Lib\site-packages\scipy\stats\_axis_nan_policy.py:430: UserWarning: `kurtosistest` p-value may be inaccurate with fewer than 20 observations; only n=14 observations were given.
  return hypotest_fun_in(*args, **kwds)
```

The Economic Condition is the most influential factor in determining academic performance (GPA), as it is statistically significant with a positive coefficient.

Other variables like Adaptability, Self Confidence, and IQ are not statistically significant at the 5% significance level, indicating they do not have a strong effect on GPA in this model.

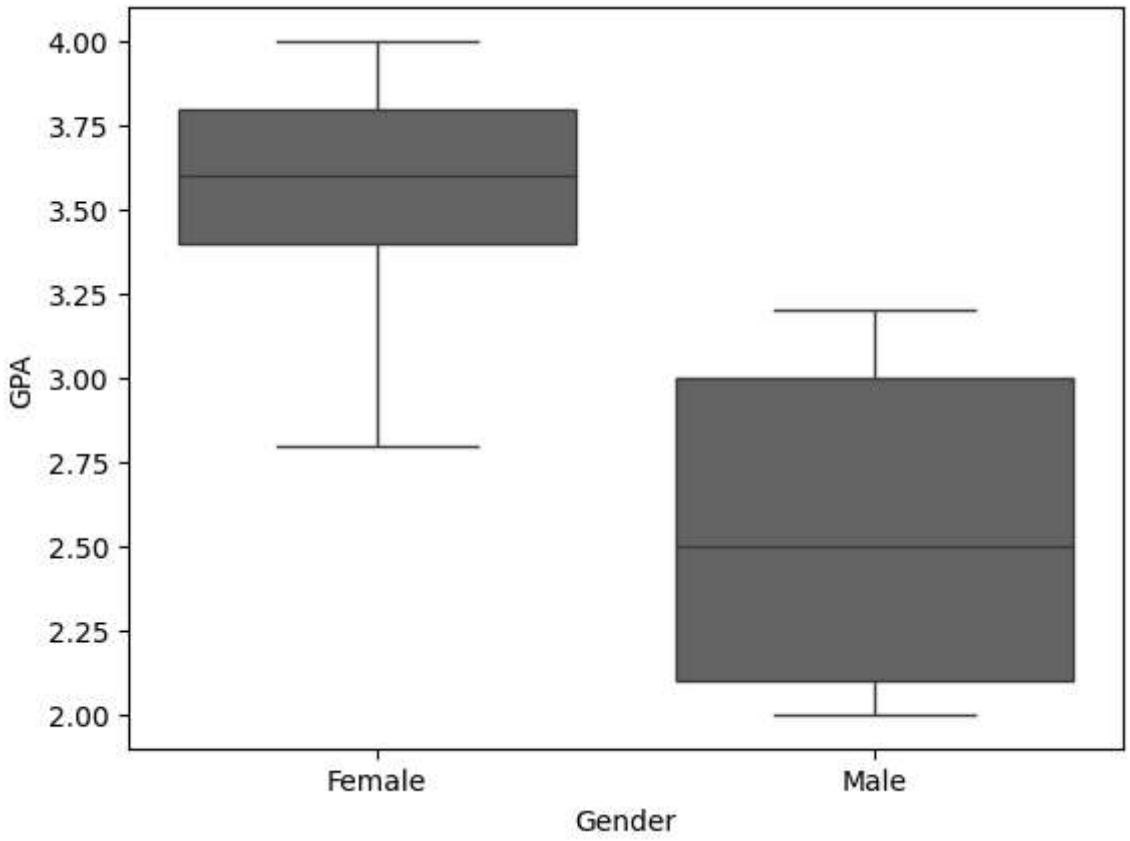
1. ii) Is there any variation of academic performance with respect to gender and economic condition.

In [66]:
```
student_df.head()
```

Out[66]:

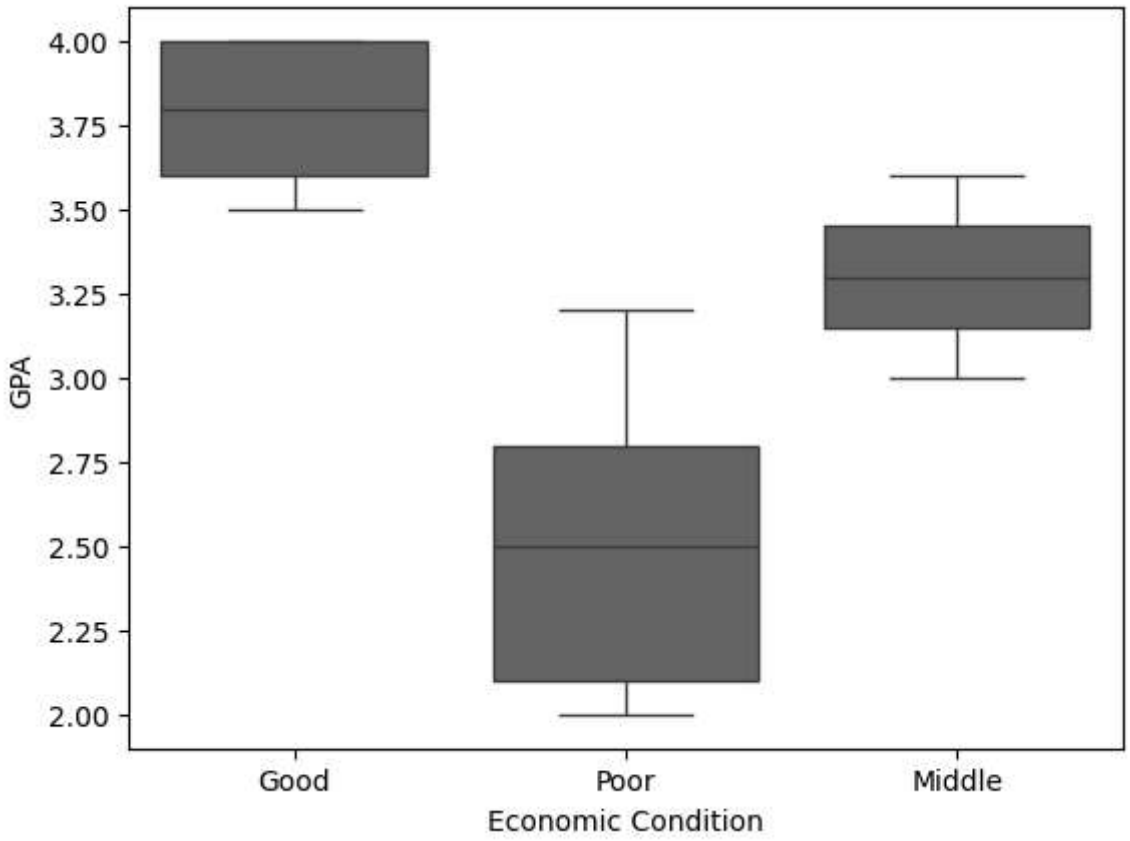
	Subject	GPA	Adaptability	Self Confidence	IQ	Gender	Economic Condition
0	1	3.8	45	60	105	Female	Good
1	2	4.0	50	10	109	Female	Good
2	3	3.2	45	50	102	Female	Poor
3	4	3.5	51	25	95	Female	Good
4	5	2.5	60	15	92	Male	Poor

In [67]:
```
#Visual test
sns.boxplot(x="Gender", y="GPA", data= student_df)
plt.show()

# T-Test
male_gpa = student_df[student_df['Gender'] == "Male"]['GPA']
female_gpa = student_df[student_df['Gender'] == "Female"]['GPA']
ttest, p_value = stats.ttest_ind(male_gpa, female_gpa)

if p_value <= 0.05:
    print("Reject Null Hypothesis. There is a difference between them")
else:
    print("Can't Reject Null Hypothesis. There is no difference")
```




```
Reject Null Hypothesis. There is a difference between them
```

In [68]:
```
# For Economic Condition
sns.boxplot(x="Economic Condition", y="GPA", data= student_df)
plt.show()

#ANOVA Test
groups = [student_df[student_df['Economic Condition'] == cond]['GPA'] for cond in s
anova_value, p_value = stats.f_oneway(*groups)

if p_value <= 0.05:
    print("Reject Null Hypothesis. There is a difference between them")
else:
    print("Can't Reject Null Hypothesis. There is no difference")
```



Reject Null Hypothesis. There is a difference between them

In [78]:
```
from statsmodels.stats.multicomp import pairwise_tukeyhsd

#Post Hoc Analysis
turkey = pairwise_tukeyhsd(student_df['GPA'], student_df_encoded['Economic Conditio
print(turkey)
```

```
Multiple Comparison of Means - Tukey HSD, FWER=0.05
===================================================
group1 group2 meandiff p-adj   lower  upper  reject
---------------------------------------------------
     1      2     0.78  0.019  0.1346 1.4254   True
     1      3     1.26 0.0004  0.6515 1.8685   True
     2      3     0.48 0.1561 -0.1654 1.1254  False
---------------------------------------------------
```

2. i) Using the best fitted model, forecast the production and consumption of tea in Bangladesh.

In [69]:
```
#Forecast Production
from sklearn.linear_model import LinearRegression

X = production_df[['Year']]
y_prod = production_df['Production (thousands of kg)']
y_cons = production_df['Consumption (in thousands of kg)']

prod_model = LinearRegression().fit(X, y_prod)
cons_model = LinearRegression().fit(X, y_cons)
```

```
future_years = pd.DataFrame({"Year": [2023, 2024, 2025]})
predict_prod = prod_model.predict(future_years)
predict_cons = cons_model.predict(future_years)

print("Predict production:", predict_prod)
print("Predict Consumption:", predict_cons)
```

```
Predict production: [102.30016317 105.88229604 109.4644289 ]
Predict Consumption: [89.06289044 91.27638695 93.48988345]
```

2. ii) Is there any significant relationship between production and consumption

In [70]:
```
#Correlation
from scipy.stats import pearsonr

corr, p_value = pearsonr(production_df['Production (thousands of kg)'], production_
print("Correlation Value is:", corr)
if p_value <= 0.05:
    print("Reject Null Hypothesis. There is a significant difference between them")
else:
    print("Can't Reject Null Hypothesis. There is no difference")
```

```
Correlation Value is: 0.4573198491523425
Can't Reject Null Hypothesis. There is no difference
```

6. 2021 Question

---

a) State the background characteristics of the respondents by displaying a percent frequency distribution table.

In [71]:
```
person_df.head()
```

Out[71]:

	ID	Age	Height (in inch)	IQ	BMI	Gender	Family Type	Smoking Habit	Disease Suffering	Psychological Stress
0	1	48	61	110	24.99	Male	Nuclear	Yes	Yes	Yes
1	2	50	62	100	20.16	Male	Nuclear	Yes	Yes	Yes
2	3	46	60	102	22.39	Male	Nuclear	Yes	No	Yes
3	4	44	57	92	20.04	Female	Nuclear	No	Yes	Yes
4	5	60	58	95	20.73	Female	Joint	No	No	Yes

In [72]:
```
cat_columns = ["Gender", "Family Type", "Smoking Habit", 'Disease Suffering', 'Psyc
for col in cat_columns:
    print(f"\nFrequency Distribution for {col}: {person_df[col].value_counts(normal
```

```
Frequency Distribution for Gender: Gender
Female    60.0
Male      40.0
Name: proportion, dtype: float64

Frequency Distribution for Family Type: Family Type
Nuclear    70.0
Joint      30.0
Name: proportion, dtype: float64

Frequency Distribution for Smoking Habit: Smoking Habit
No     75.0
Yes    25.0
Name: proportion, dtype: float64

Frequency Distribution for Disease Suffering: Disease Suffering
No     60.0
Yes    40.0
Name: proportion, dtype: float64

Frequency Distribution for Psychological Stress: Psychological Stress
Yes    55.0
No     45.0
Name: proportion, dtype: float64
```

b) Explore the influential factors affecting on IQ of the respondents by applying multiple linear regression model.¶

In [73]:
```
y = person_df['IQ']

# Independent variable
df2 = person_df.copy()
df2['Smoking Habit'] = df2['Smoking Habit'].map({"Yes": 1, "No": 0})
df2['Disease Suffering'] = df2['Disease Suffering'].map({"Yes": 1, "No": 0})
df2['Psychological Stress'] = df2['Psychological Stress'].map({"Yes": 1, "No": 0})
X = df2[['Age', 'Height (in inch)', 'BMI', 'Smoking Habit', 'Disease Suffering', 'P

X = sm.add_constant(X)
model = sm.OLS(y, X).fit()
print(model.summary())
```

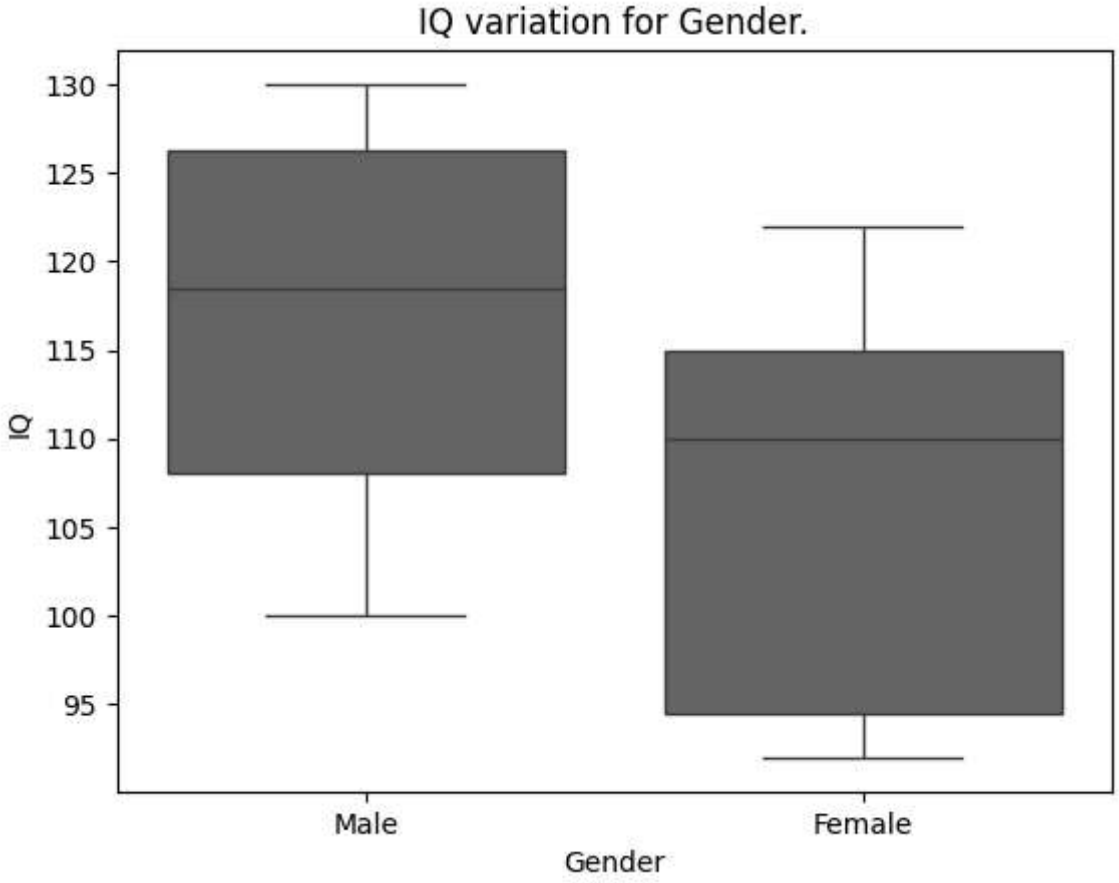
```
                            OLS Regression Results
==============================================================================
Dep. Variable:                     IQ   R-squared:                       0.612
Model:                            OLS   Adj. R-squared:                  0.432
Method:                 Least Squares   F-statistic:                     3.413
Date:                Wed, 09 Apr 2025   Prob (F-statistic):             0.0301
Time:                        16:33:32   Log-Likelihood:                -68.163
No. Observations:                  20   AIC:                             150.3
Df Residuals:                      13   BIC:                             157.3
Df Model:                           6
Covariance Type:            nonrobust
================================================================================
                   coef    std err          t      P>|t|      [0.025      0.
975]
--------------------------------------------------------------------------------
const           66.5889     28.197      2.362      0.034       5.674     12
7.504
Age              0.2699      0.366      0.738      0.473      -0.520
1.060
Height (in inch)    0.1175      0.202      0.583      0.570      -0.318
0.553
BMI              1.1593      0.519      2.232      0.044       0.037
2.281
Smoking Habit   12.4733      6.052      2.061      0.060      -0.602      2
5.549
Disease Suffering    1.2773      5.282      0.242      0.813     -10.133      1
2.687
Psychological Stress   -15.4266      6.500     -2.373      0.034     -29.469      -
1.384
==============================================================================
Omnibus:                        1.554   Durbin-Watson:                   1.164
Prob(Omnibus):                  0.460   Jarque-Bera (JB):                1.276
Skew:                           0.455   Prob(JB):                        0.528
Kurtosis:                       2.160   Cond. No.                     1.19e+03
==============================================================================

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly spe
cified.
[2] The condition number is large, 1.19e+03. This might indicate that there are
strong multicollinearity or other numerical problems.
```

c) Determine the variation of IQ with respect to background characteristic.

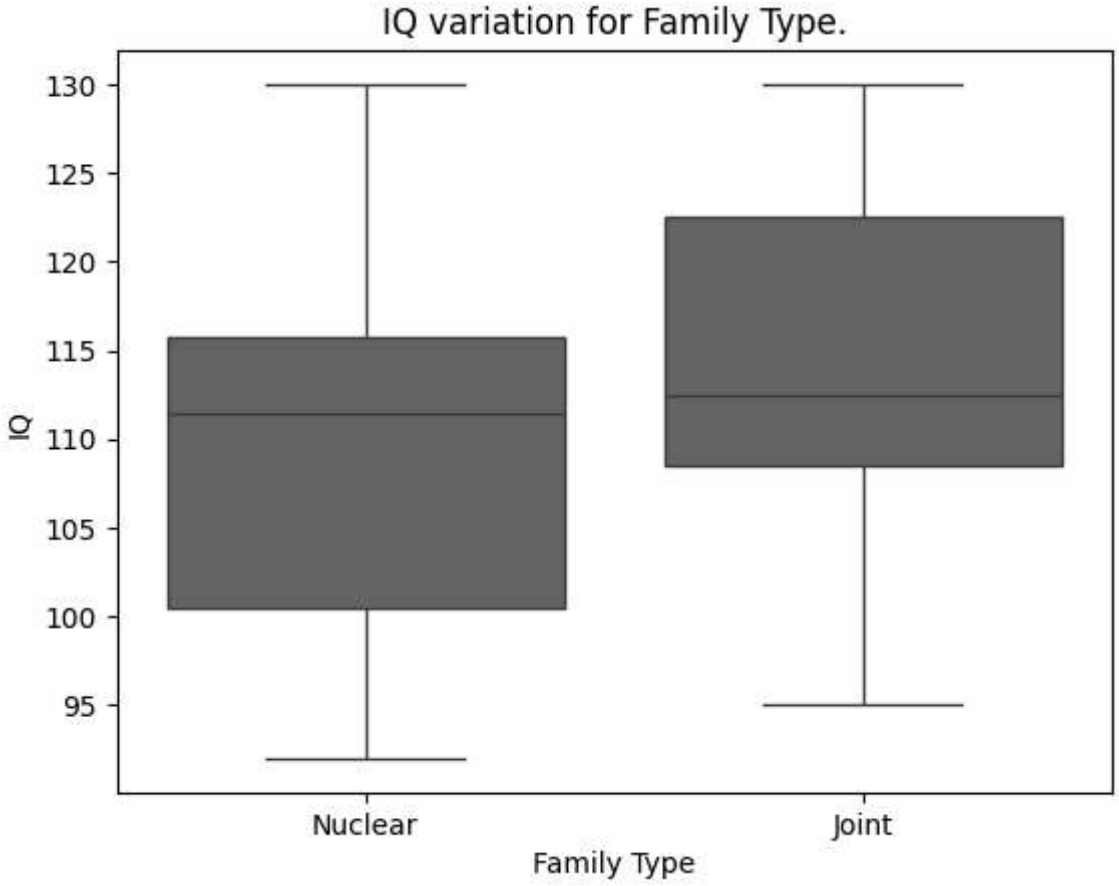
In [74]:
```
#Visulization
cat_columns = ["Gender", "Family Type", "Smoking Habit", 'Disease Suffering', 'Psyc

for col in cat_columns:
    print("\nVariation for:", col)
    sns.boxplot(x=col, y="IQ", data= person_df)
    plt.title(f"IQ variation for {col}.")
```

```
    plt.show()

    print(person_df.groupby(col)['IQ'].describe())
```

```
Variation for: Gender
```

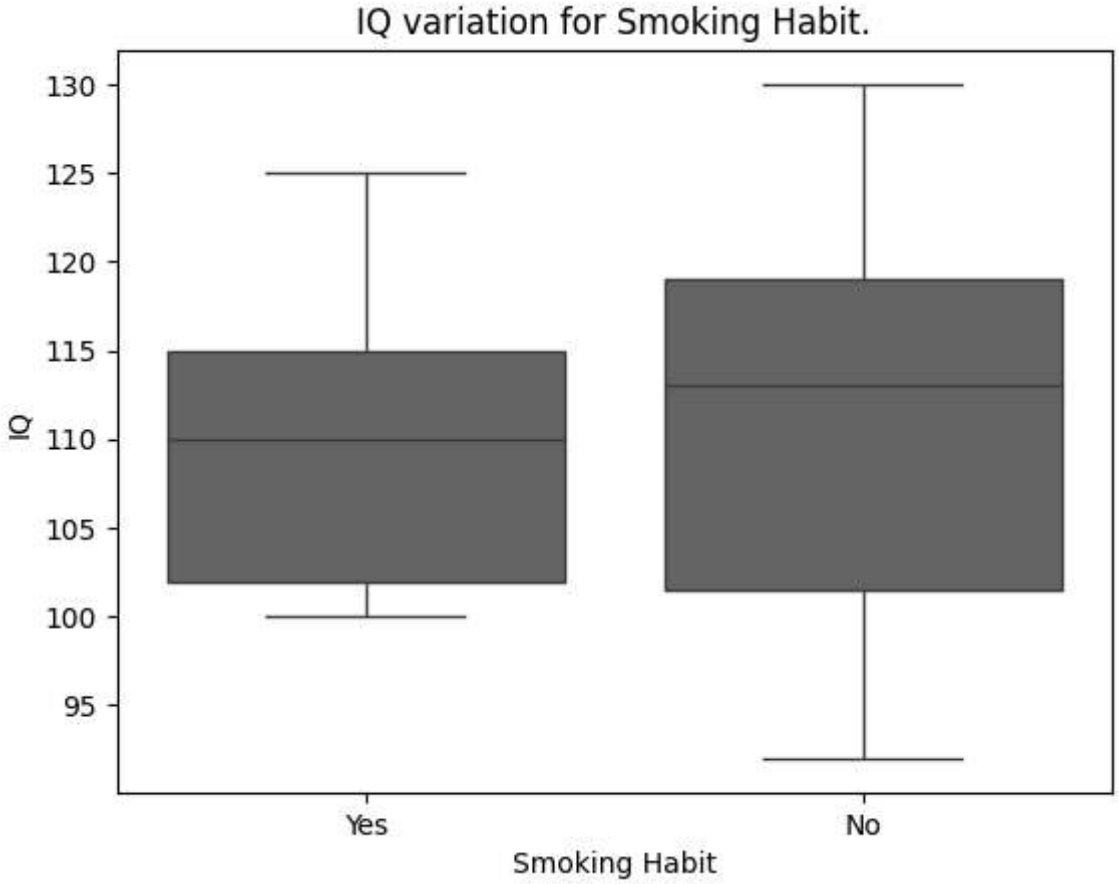



```
        count    mean        std    min    25%    50%     75%    max
Gender
Female   12.0  106.75  10.779821   92.0   94.5  110.0  115.00  122.0
Male      8.0  116.75  11.913378  100.0  108.0  118.5  126.25  130.0

Variation for: Family Type
```



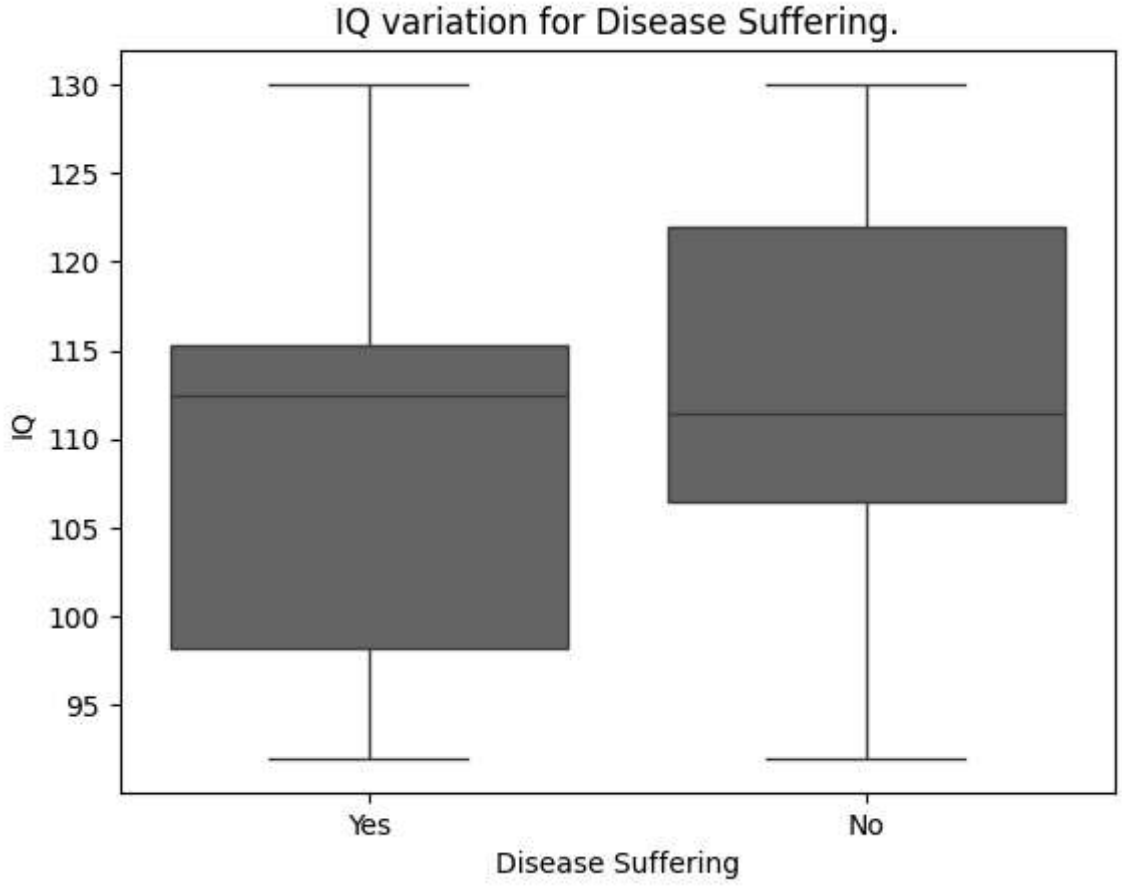
Family Type	count	mean	std	min	25%	50%	75%	max
Joint	6.0	113.833333	12.576433	95.0	108.5	112.5	122.50	130.0
Nuclear	14.0	109.428571	12.023786	92.0	100.5	111.5	115.75	130.0

Variation for: Smoking Habit



	count	mean	std	min	25%	50%	75%	max
Smoking Habit								
No	15.0	110.866667	12.922111	92.0	101.5	113.0	119.0	130.0
Yes	5.0	110.400000	10.163661	100.0	102.0	110.0	115.0	125.0

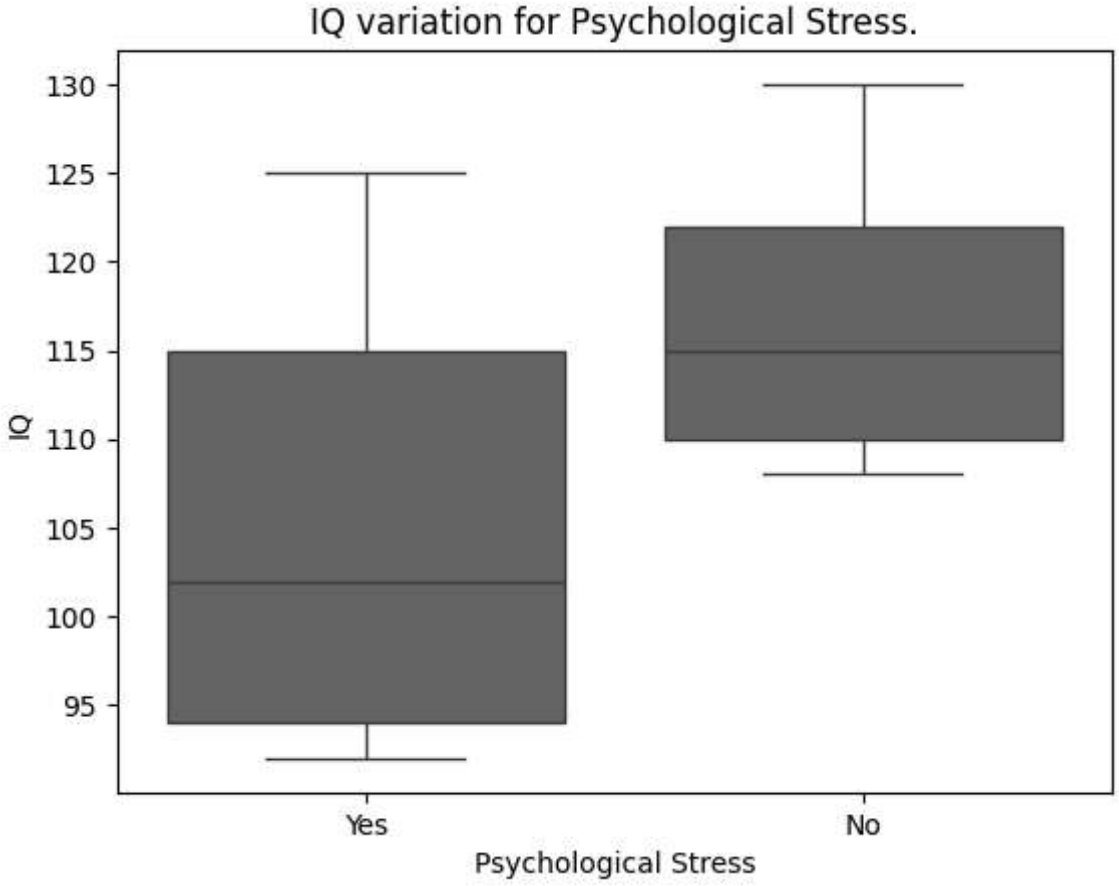
Variation for: Disease Suffering



	count	mean	std	min	25%	50%	75%	\
Disease Suffering								
No	12.0	112.000	11.739599	92.0	106.50	111.5	122.00	
Yes	8.0	108.875	13.032240	92.0	98.25	112.5	115.25	

	max
Disease Suffering	
No	130.0
Yes	130.0

Variation for: Psychological Stress



	count	mean	std	min	25%	50%	\
Psychological Stress							
No	9.0	117.777778	8.526104	108.0	110.0	115.0	
Yes	11.0	105.000000	11.670476	92.0	94.0	102.0	

	75%	max
Psychological Stress		
No	122.0	130.0
Yes	115.0	125.0

d) Determine the factor affecting on psychological stress of the respondents.

In [75]:
```
person_df.head()
```

Out[75]:

	ID	Age	Height (in inch)	IQ	BMI	Gender	Family Type	Smoking Habit	Disease Suffering	Psychological Stress
0	1	48	61	110	24.99	Male	Nuclear	Yes	Yes	Yes
1	2	50	62	100	20.16	Male	Nuclear	Yes	Yes	Yes
2	3	46	60	102	22.39	Male	Nuclear	Yes	No	Yes
3	4	44	57	92	20.04	Female	Nuclear	No	Yes	Yes
4	5	60	58	95	20.73	Female	Joint	No	No	Yes

In [76]:
```
# Independent variable
df2 = person_df.copy()
df2['Smoking Habit'] = df2['Smoking Habit'].map({"Yes": 1, "No": 0})
df2['Disease Suffering'] = df2['Disease Suffering'].map({"Yes": 1, "No": 0})
df2['Psychological Stress'] = df2['Psychological Stress'].map({"Yes": 1, "No": 0})

y = df2['Psychological Stress']
X = df2[['Age', 'Height (in inch)', 'BMI',  'IQ']]
X = sm.add_constant(X)

model = sm.Logit(y, X).fit()
print(model.summary())
```

```
Optimization terminated successfully.
         Current function value: 0.472440
         Iterations 8
                           Logit Regression Results
==============================================================================
Dep. Variable:     Psychological Stress   No. Observations:                   20
Model:                            Logit   Df Residuals:                       15
Method:                             MLE   Df Model:                            4
Date:                  Wed, 09 Apr 2025   Pseudo R-squ.:                  0.3135
Time:                          16:33:32   Log-Likelihood:                -9.4488
converged:                         True   LL-Null:                       -13.763
Covariance Type:             nonrobust   LLR p-value:                   0.07110
==============================================================================
                     coef    std err          z      P>|z|      [0.025      0.975]
------------------------------------------------------------------------------
const             17.6081     10.685      1.648      0.099      -3.334      38.550
Age                0.0223      0.093      0.240      0.811      -0.160       0.204
Height (in inch)  -0.1017      0.195     -0.522      0.602      -0.484       0.280
BMI               -0.1141      0.262     -0.435      0.663      -0.628       0.400
IQ                -0.0816      0.118     -0.690      0.490      -0.314       0.150
==============================================================================
```