Tarea Modulo Estadistica

Esta tarea pertenece al modulo de **Estadistica** del Máster de Big Data y Data Science por la Universidad Complutense de Madrid

In [1]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt

df = pd.read_excel('Descripcion/ARCHIVODATOSEVALUACION24.xlsx')

# Muestra las primeras filas del archivo para verificar que se cargó correctamente
print(df.head())
df.info()
df.describe()
```

```
   Grupo de control  Nivel glucosa basal  Nivel glucosa 60 min
0                 1                   90                   136
1                 1                   82                   151
2                 1                   80                   148
3                 1                   75                   138
4                 1                   74                   141
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 65 entries, 0 to 64
Data columns (total 3 columns):
 #   Column                Non-Null Count  Dtype
---  ------                --------------  -----
 0   Grupo de control      65 non-null     int64
 1   Nivel glucosa basal   65 non-null     int64
 2   Nivel glucosa 60 min  65 non-null     int64
dtypes: int64(3)
memory usage: 1.7 KB
```

Out[1]:

	Grupo de control	Nivel glucosa basal	Nivel glucosa 60 min
count	65.000000	65.000000	65.000000
mean	1.461538	87.707692	159.630769
std	0.502398	9.363698	16.701955
min	1.000000	65.000000	131.000000
25%	1.000000	80.000000	147.000000
50%	1.000000	88.000000	158.000000
75%	2.000000	96.000000	172.000000
max	2.000000	106.000000	198.000000

EJERCICIO 1

A) Obtener, usando algún programa estadístico, las medidas de centralización y dispersión para cada uno de los dos grupos de control para el nivel de glucosa basal, especificando para cada uno de los casos si la media es o no representativa.

Medidas de Centralizacion: (Separando los grupos)

In [2]:
```
adultos = df[df["Grupo de control"]==2]
jovenes = df[df["Grupo de control"]==1]
```

In [3]:
```
media_adultos=round(adultos["Nivel glucosa basal"].mean(),2)
mediana_adultos=round(adultos["Nivel glucosa basal"].median(),2)
moda_adultos=round(adultos["Nivel glucosa basal"].mode(),2)

media_jovenes=round(jovenes["Nivel glucosa basal"].mean(),2)
mediana_jovenes=round(jovenes["Nivel glucosa basal"].median(),2)
moda_jovenes=round(jovenes["Nivel glucosa basal"].mode(),2)

desviacion_estandar_adultos= round(adultos["Nivel glucosa basal"].std(),2)
desviacion_estandar_jovenes= round(jovenes["Nivel glucosa basal"].std(),2)

coeficiente_variacion_adultos= desviacion_estandar_adultos / media_adultos
coeficiente_variacion_jovenes= desviacion_estandar_jovenes / media_jovenes
```

In [4]:
```
print ("Grupo Adultos (grupo de control 2):")
print ("Media de Adultos= " ,media_adultos)
print ("Mediana de Adultos= " ,mediana_adultos)
print ("Moda de Adultos= ", moda_adultos)

print("")

print ("Grupo Jovenes (grupo de control 1):")
print ("Media de Jovenes= " ,media_jovenes)
print ("Mediana de Jovenes= " ,mediana_jovenes)
print ("Moda DE jOVENES= ", moda_jovenes)

print("")

print ("Desviacion estandar Adultos= ", desviacion_estandar_adultos)
print ("Desviacion estandar Jovenes= ", desviacion_estandar_jovenes)

print("")

print("Coeficiente de variación del grupo de Adultos (grupo de control 2)= ", round
print("Coeficiente de variación del grupo de Jóvenes (grupo de control 1)= ", round
```

```
Grupo Adultos (grupo de control 2):
Media de Adultos=  90.83
Mediana de Adultos=  90.5
Moda de Adultos=  0    88
Name: Nivel glucosa basal, dtype: int64

Grupo Jovenes (grupo de control 1):
Media de Jovenes=  85.03
Mediana de Jovenes=  82.0
Moda DE jOVENES=  0    75
1    79
2    82
3    90
Name: Nivel glucosa basal, dtype: int64

Desviacion estandar Adultos=  8.46
Desviacion estandar Jovenes=  9.38

Coeficiente de variación del grupo de Adultos (grupo de control 2)=  9.31 %
Coeficiente de variación del grupo de Jóvenes (grupo de control 1)=  11.03 %
```

Si nos basamos en la tabla que se muestra acontinuacion, la cual muestra los gradons en los que la media es representativa o no, podemos saber que tan representativa es la media tomando en cuenta los resultados obtenidos.

Valor del coeficiente de variabilidad	Grado en que la media representa a la serie
De 0 a menos del 10 %	La media es altamente representativa
De 10 a menos del 20 %	La media tiene representatividad.
De 20 a menos del 30 %	La media tiene representatividad
De 30 a menos del 40 %	La media tiene representación dudosa.
De 40 % o más	La media carece de representatividad.

Grupo de control 1 (Jovenes)= 11.03%

Grupo de control 2 (Adultos)= 9.31%

Podemos deducir que para ambos casos la media es **ALTA** mente representativa debido que para el grupo de los adultos el coeficiente de variacion se encuentra entre el rango del 0-10%, incluso se puede deducir que para los jovenes la media tambien es altamente representativa debido a que el coeficiente de variacion esta muy cercano al 10% solo superandolo por un 1% lo cual es despreciable.

Medidas de Dispersion

In [5]:
```
# Rango
rango_adultos = max(adultos['Nivel glucosa basal']) - min(adultos['Nivel glucosa ba
rango_jovenes = max(jovenes['Nivel glucosa basal']) - min(jovenes['Nivel glucosa ba

# Varianza
varianza_adultos = round(adultos['Nivel glucosa basal'].var(), 2)  # Varianza adult
```

```
varianza_jovenes = round(jovenes['Nivel glucosa basal'].var(), 2)  # Varianza jóven


# Imprimir resultados
print('Rango')
print(f'El rango del grupo de adultos es: {rango_adultos}')
print(f'El rango del grupo de jóvenes es: {rango_jovenes}')

print()

print('Varianza')
print(f'La varianza del grupo de adultos es {varianza_adultos}')
print(f'La varianza del grupo de jóvenes es {varianza_jovenes}')
```

```
Rango
El rango del grupo de adultos es: 29
El rango del grupo de jóvenes es: 39

Varianza
La varianza del grupo de adultos es 71.59
La varianza del grupo de jóvenes es 87.97
```

B) Estudiar la simetría y la curtosis del nivel de glucosa basal en los adultos (grupo de control 2).

In [6]:
```
simetria_adultos=round(adultos["Nivel glucosa basal"].skew(),2)

curtosis_adultos=round(adultos["Nivel glucosa basal"].kurt(),2)
```

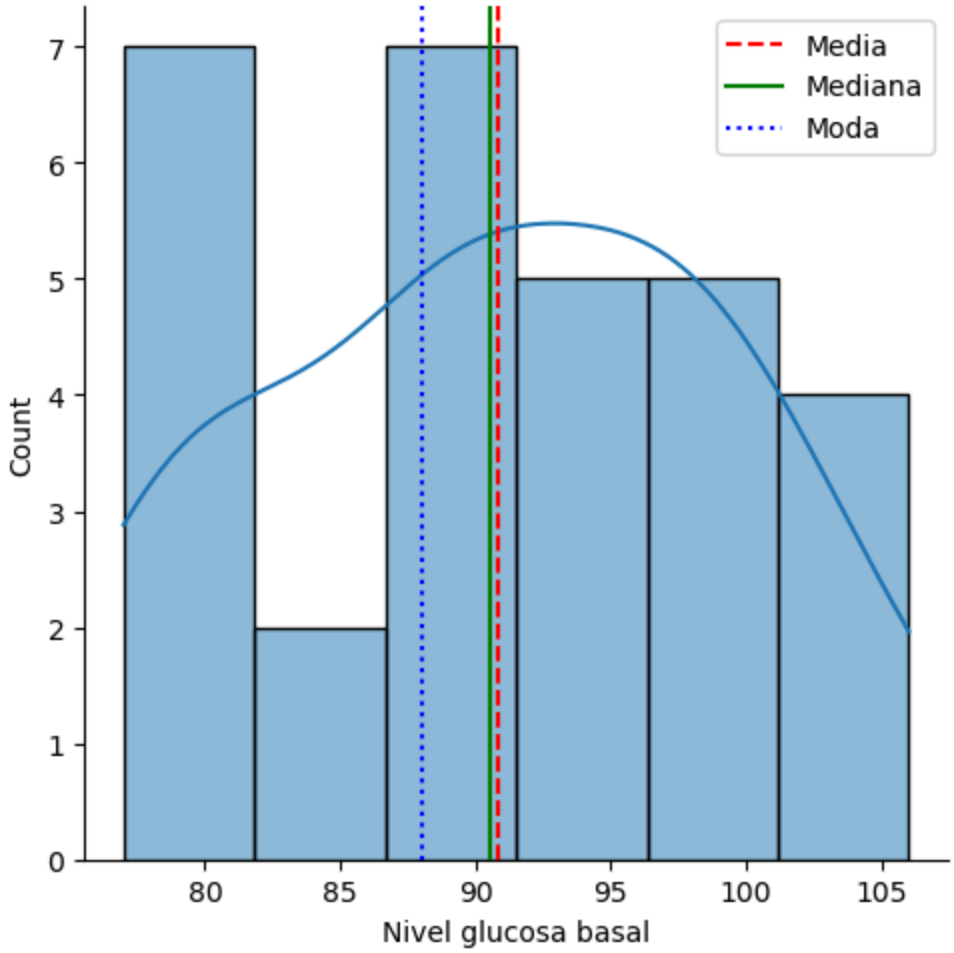
In [7]:
```
print("simetria para el grupo de control 2 (Adultos)= ", simetria_adultos)

print()

print("curtosis para el grupo de control 2 (Adultos)= ", curtosis_adultos)
```

```
simetria para el grupo de control 2 (Adultos)=  -0.08

curtosis para el grupo de control 2 (Adultos)=  -1.03
```

In [8]:
```
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt

# Filtrar datos
adultos = df[df["Grupo de control"] == 2]

# Calcular media, mediana y moda
media_adultos = round(adultos["Nivel glucosa basal"].mean(), 2)
mediana_adultos = round(adultos["Nivel glucosa basal"].median(), 2)
moda_adultos = round(adultos["Nivel glucosa basal"].mode()[0], 2)  # Obtener el pri

# Crear un DataFrame para la tabla
tabla_estadisticas = pd.DataFrame({
    'Estadística': ['Media', 'Mediana', 'Moda'],
    'Valor': [media_adultos, mediana_adultos, moda_adultos]
})
```

```
# Mostrar la tabla
print(tabla_estadisticas)

# Crear el gráfico de distribución
sns.displot(data=adultos, x="Nivel glucosa basal", kde=True)

# Agregar líneas para la media, mediana y moda
plt.axvline(media_adultos, color='r', linestyle='--', label='Media')
plt.axvline(mediana_adultos, color='g', linestyle='-', label='Mediana')
plt.axvline(moda_adultos, color='b', linestyle=':', label='Moda')

# Agregar leyenda
plt.legend()

# Mostrar el gráfico
plt.show()
```

```
  Estadística  Valor
0       Media  90.83
1     Mediana  90.50
2        Moda  88.00
```



Estudio de la simetria y la curtosis

El histograma parece estar ligeramente torcido hacia la derecha, lo que indica que no es simetrica, da a entender una asimetría positiva. lo cual indica que esta mas inclinado hacia el

lado derecho que el izquierdo. La distribución parece tener un pico alrededor de la moda y va disminuyendo gradualmente a ambos lados. No se aprecia la presencia de colas pesadas ni valores atípicos extremos, lo que indica que la curtosis es probablemente similar a la de una distribución normal mesocúrtica.

Podemos resumir que la distribución de los niveles de glucosa basal es ligeramente asimétrica positiva y probablemente mesocúrtica.

C) Indicar para cada una de las variables de estudio (nivel glucosa basal y nivel glucosa pasados 60 min) y en el grupo de control 1 el valor de los cuartiles y su significado y obtener el box- plot (diagrama de cajas) correspondiente. Estudiar la presencia de valores atípicos.

Obtenemos los cuartiles:

Los cuartiles son valores que dividen una muestra de datos en 4 partes iguales :

-Q1: El primer cuartil Q1, el 25% de los datos es menor o igual a este valor

-Q2: El segundo cuartil Q2 , el valor que divide el 50% de los datos, es también la **mediana**.

-Q3: El tercer cuartil Q3, el 75% de los datos es menor o igual a este valor

In [9]:
```
import numpy as np
q1_nivel_glucosa_basal= np.percentile(jovenes["Nivel glucosa basal"], 25)
print ("Primer cuartil (Q1)", q1_nivel_glucosa_basal)
q1_pasado_60_min= np.percentile(jovenes["Nivel glucosa 60 min"], 25)
print ("primer cuartil pasado 60 min(Q1)", q1_pasado_60_min)

print()

q2_nivel_glucosa_basal= np.percentile(jovenes["Nivel glucosa basal"], 50)
print ("Segundo cuartil (Q2)", q2_nivel_glucosa_basal)
q2_pasado_60_min= np.percentile(jovenes["Nivel glucosa 60 min"], 50)
print ("Segundo cuartil pasado 60 min(Q2)", q2_pasado_60_min)

print()

q3_nivel_glucosa_basal= np.percentile(jovenes["Nivel glucosa basal"], 75)
print ("Tercer cuartil (Q3)", q3_nivel_glucosa_basal)
q3_pasado_60_min= np.percentile(jovenes["Nivel glucosa 60 min"], 75)
print ("Tercer cuartil pasado 60 min(Q3)", q3_pasado_60_min)

print()
```

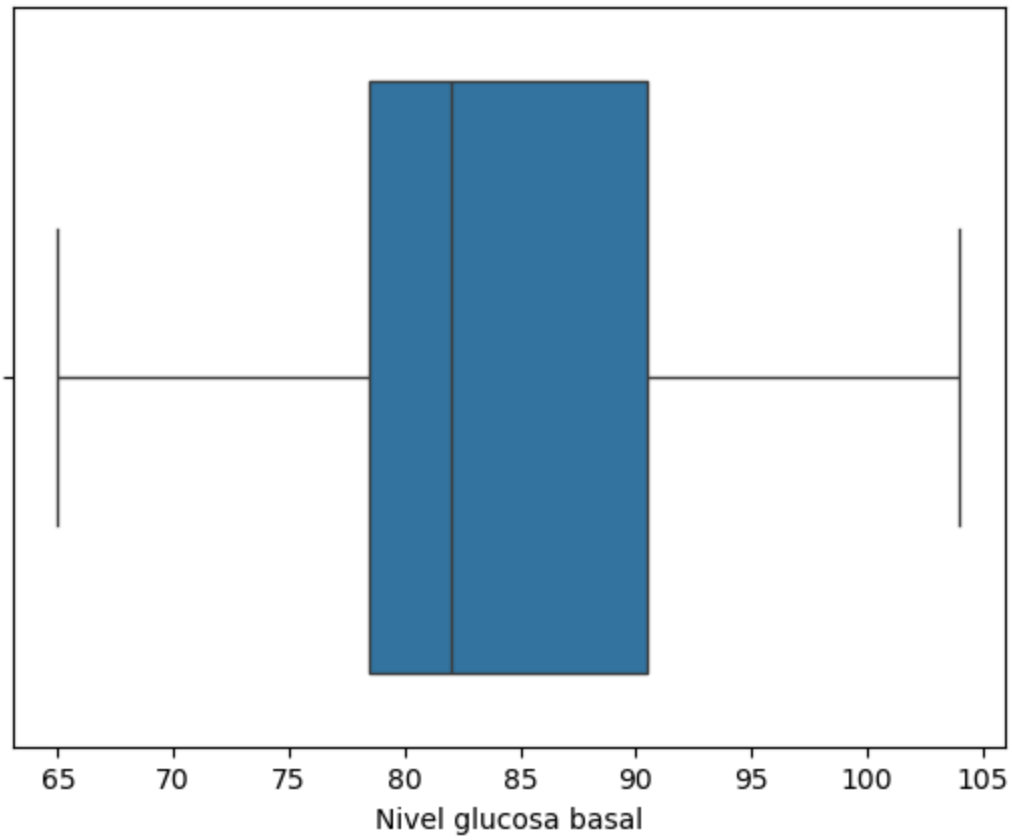
```
Primer cuartil (Q1) 78.5
primer cuartil pasado 60 min(Q1) 141.0

Segundo cuartil (Q2) 82.0
Segundo cuartil pasado 60 min(Q2) 148.0

Tercer cuartil (Q3) 90.5
Tercer cuartil pasado 60 min(Q3) 153.0
```

box-plot (diagrama de cajas):

Nivel de glucosa basal para el grupo de control 1 (jovenes):

In [10]:
```
sns.boxplot(x=jovenes["Nivel glucosa basal"])
```

Out[10]:
```
<Axes: xlabel='Nivel glucosa basal'>
```

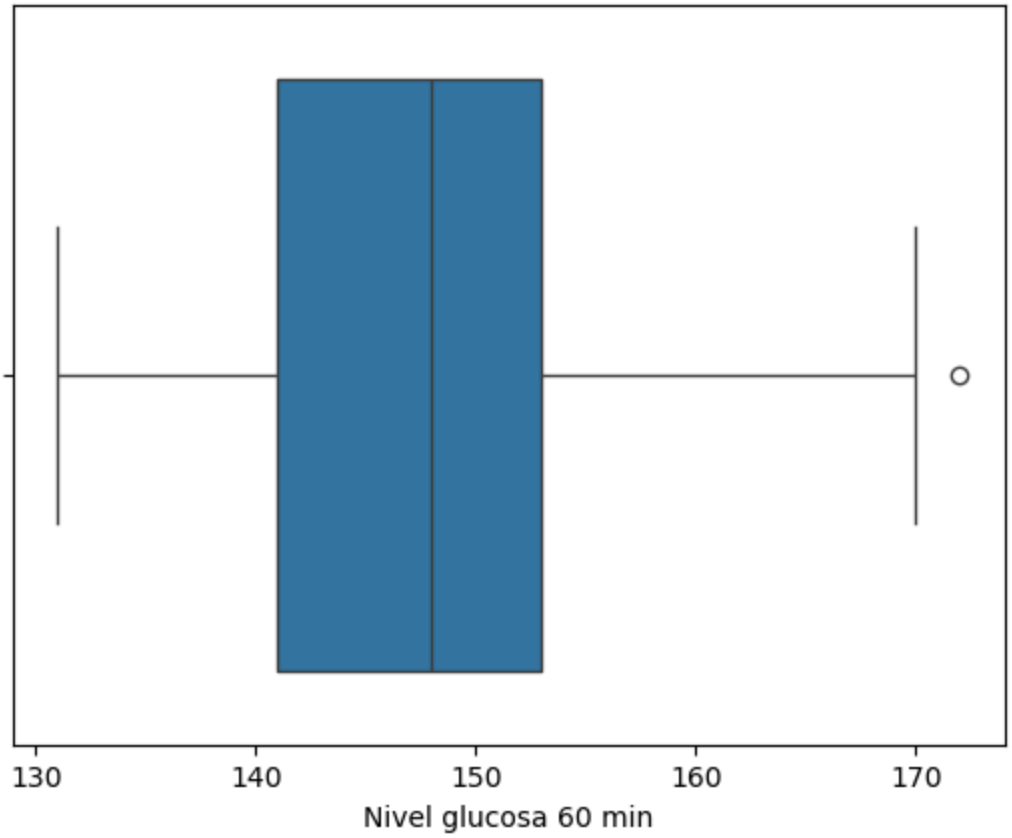


En el diagrama de cajas mostrado anteriormente, "nivel de glucosa basal para el grupo de personas jovenes", **no se aprecian la existencia de valores atipicos** es decir ,fuera del límite superior e inferior del diagrama de caja y bigote.

In [11]:
```
import seaborn as sns

# Generar el boxplot para visualizar
sns.boxplot(x=jovenes["Nivel glucosa 60 min"])
```

```python
# Calcular los cuartiles y el IQR
Q1 = jovenes["Nivel glucosa 60 min"].quantile(0.25)
Q3 = jovenes["Nivel glucosa 60 min"].quantile(0.75)
IQR = Q3 - Q1

# Calcular los límites de los valores atípicos
limite_inferior = Q1 - 1.5 * IQR
limite_superior = Q3 + 1.5 * IQR

# Filtrar los valores que están fuera de estos límites
valores_atipicos_60_min = jovenes[(jovenes["Nivel glucosa 60 min"] < limite_inferio

# Imprimir los valores atípicos
print("Valores atípicos en 'Nivel glucosa 60 min':")
print(valores_atipicos_60_min["Nivel glucosa 60 min"])
```

```
Valores atípicos en 'Nivel glucosa 60 min':
6    172
Name: Nivel glucosa 60 min, dtype: int64
```

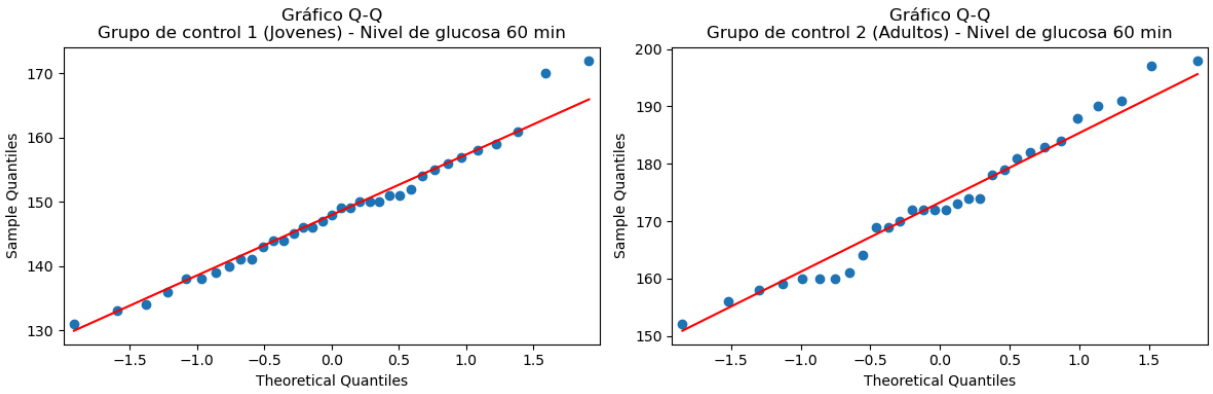


Para el grafico que se muestra anteriormente,"Nivel de glucosa pasado 60 min para el grupo de personas jovenes", si que se muestra la existencia de un valor atipico o outlier, es decir que se encuentra fuera del limite superior del diagrama el cual es de 170, para determinar cual es el valor de dicho outlier primero se procedio a sacar el El Rango Intercuartílico (IQR), luego se calculo el limite superior e inferior para finalmente filtrar los valores que se encuentran fuera de dicho limite. Obteniendo el valor atipico el cual es 172.

D) Estudiar la normalidad de los datos de cada uno de los grupos de control estudiados para el nivel de glucosa pasados 60 minutos.

Para estudiar la normalidad de cada uno de los grupos de control (Jovenes y adultos) para el Nivel de glucosa pasados 60 min utilizaremos una grafica Q-Q Plot:

In [12]:
```
import statsmodels.api as sm
import matplotlib.pyplot as plt

# Crear el gráfico Q-Q
fig, axs = plt.subplots(1, 2, figsize=(12, 4))

# Q-Q plot para "Nivel de glucosa 60 min" en GC1
sm.graphics.qqplot(jovenes['Nivel glucosa 60 min'], line='s', ax=axs[0])
axs[0].set_title('Gráfico Q-Q\nGrupo de control 1 (Jovenes) - Nivel de glucosa 60 m

# Q-Q plot para "Nivel de glucosa 60 min" en GC2
sm.graphics.qqplot(adultos['Nivel glucosa 60 min'], line='s', ax=axs[1])
axs[1].set_title('Gráfico Q-Q\nGrupo de control 2 (Adultos) - Nivel de glucosa 60 m

# Ajustar el layout y mostrar
fig.tight_layout()
plt.show()
```



La línea roja en cada gráfico representa la distribución normal teórica. Si los puntos azules se alinean cerca de esta línea, indica que los datos siguen una distribución normal.

Para el **grupo de control 1 (Jovenes):** La mayoría de los puntos se alinean bien con la línea roja de referencia, lo que da a entender que los datos siguen una distribución aproximadamente normal. Sin embargo, hay algunas desviaciones en los extremos, lo que podría indicar una desviación de la normalidad en los extremos de la distribución. Para el **grupo de control 2 (Adultos):** Los puntos muestran una mayor desviación de la línea roja de referencia, especialmente en los extremos. Esto sugiere que para dichos datos del no siguen una distribución normal tan de cerca como en el grupo de jóvenes. Las desviaciones en los extremos indican que podría haber una distribución sesgada.

Para complementar el análisis visual, puedes realizar pruebas estadísticas de normalidad, como: Prueba de **Shapiro-Wilk:** Evalúa la hipótesis de que los datos provienen de una distribución normal:

In [13]:
```
from scipy import stats

# Prueba de normalidad Shapiro-Wilk para jóvenes
shapiro_jovenes = stats.shapiro(jovenes['Nivel glucosa 60 min'])
print('Prueba de Shapiro-Wilk para Jóvenes:')
print(f'Estadístico: {shapiro_jovenes.statistic}, p-valor: {shapiro_jovenes.pvalue}

print()

# Prueba de normalidad Shapiro-Wilk para adultos
shapiro_adultos = stats.shapiro(adultos['Nivel glucosa 60 min'])
print('Prueba de Shapiro-Wilk para Adultos:')
print(f'Estadístico: {shapiro_adultos.statistic}, p-valor: {shapiro_adultos.pvalue}
```

```
Prueba de Shapiro-Wilk para Jóvenes:
Estadístico: 0.9735987272599481, p-valor: 0.5493438105627045

Prueba de Shapiro-Wilk para Adultos:
Estadístico: 0.9659754286699488, p-valor: 0.4356766397527704
```

Si el p-valor < 0.05: Se rechaza la hipótesis nula de normalidad, lo que indica que los datos no siguen una distribución normal.

Si el p-valor ≥ 0.05: No se rechaza la hipótesis nula de normalidad, lo que sugiere que los datos pueden considerarse normalmente distribuidos.

Como podemos apreciar en los resultados obtenidos para ambos casos:

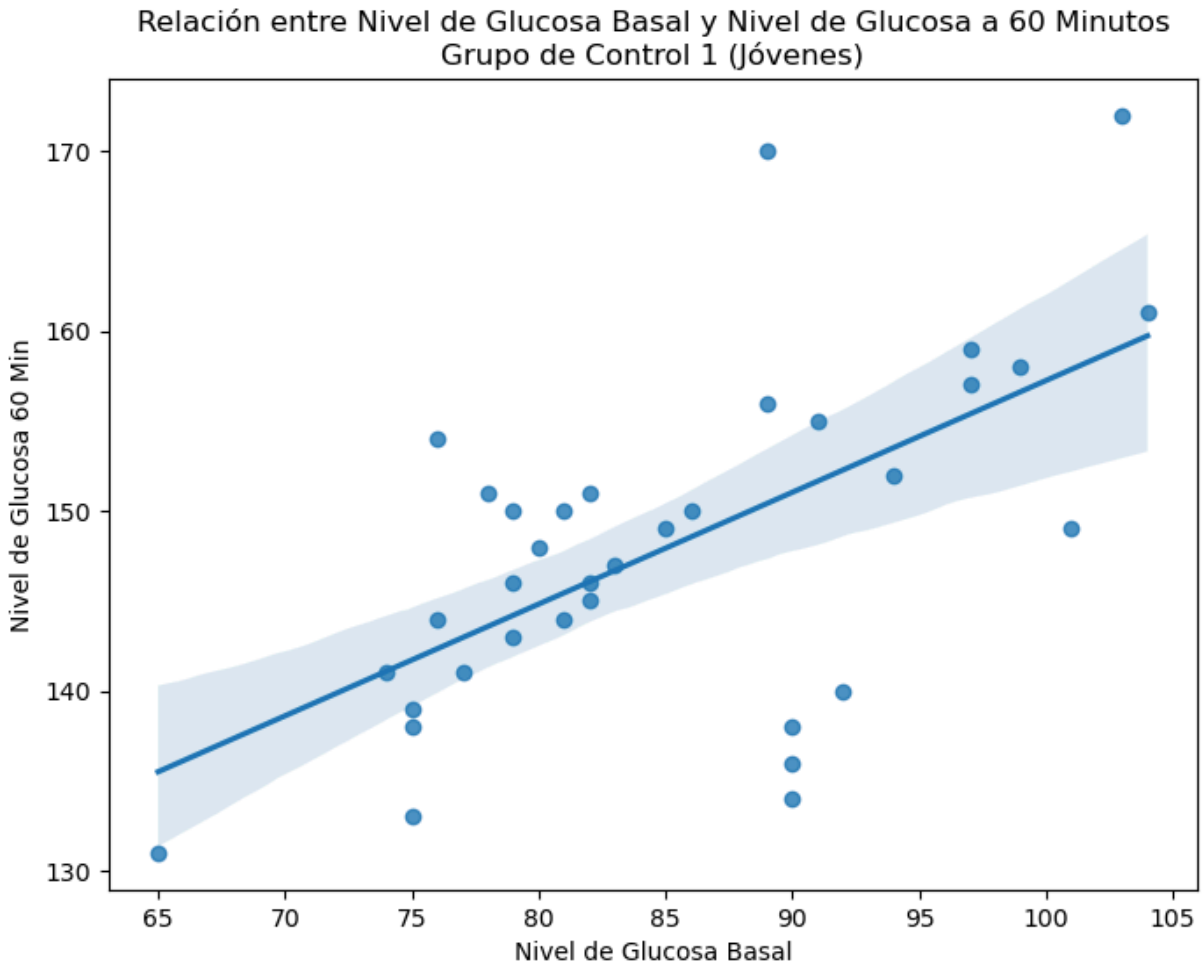
Jovenes: P-valor: 0.54>0.05 y para el grupo de **Adultos:** P-valor: 0.43>0.05

Podemos ver que en ambos casos el P-valor es superior a los 0.05. Por lo tanto podemos afirmar que nuestros datos se distribuyen siguiendo una **distribución normal.**

EJERCICIO 2

a) Estudiar la relación lineal existente entre estas dos variables de estudio gráficamente y mediante algún valor estadístico de forma razonada.

In [14]:
```
import seaborn as sns
import matplotlib.pyplot as plt

# Crear gráfico de dispersión con línea de regresión
plt.figure(figsize=(8, 6))
sns.regplot(data=jovenes, x='Nivel glucosa basal', y='Nivel glucosa 60 min', ci=95)
plt.title('Relación entre Nivel de Glucosa Basal y Nivel de Glucosa a 60 Minutos\nG
plt.xlabel('Nivel de Glucosa Basal')
plt.ylabel('Nivel de Glucosa 60 Min')
plt.show()
```



In [15]:
```
# Calcular el coeficiente de correlación de Pearson
correlacion_pearson = jovenes['Nivel glucosa basal'].corr(jovenes['Nivel glucosa 60

print(f"Coeficiente de correlación de Pearson (Grupo de Control 1 - Jóvenes): {corr
```

```
Coeficiente de correlación de Pearson (Grupo de Control 1 - Jóvenes): 0.61
```

Para el caso de ambas variables, Nivel de glucosa basal y nivel de glucosa basal pasado 60 min, en el grupo de control 1 (Jovenes) podemos concluir que existe una **corelacion positiva**. Esto se puede apreciar en el diagrama de dispersion y tambien por el resultado obtedido en el coeficiente de correlacion de Pearson (0.61).

B) Obtener un modelo lineal que explica el nivel de glucosa en sangre a los 60 minutos en función del nivel basal del paciente y realizar la estimación para un paciente cuyo nivel basal es 83 mg/Dl

In [16]:
```
import statsmodels.api as sm

# Definir la variable independiente (Nivel glucosa basal) y dependiente (Nivel gluc
X = jovenes['Nivel glucosa basal']
y = jovenes['Nivel glucosa 60 min']

# Añadir una constante a X para incluir el intercepto en el modelo
X = sm.add_constant(X)
```

```
# Crear el modelo de regresión lineal y ajustarlo
modelo = sm.OLS(y, X).fit()

# Mostrar el resumen del modelo
print(modelo.summary())

# Realizar la estimación para un paciente con nivel basal de 83 mg/dL
nivel_basal_nuevo = 83
X_nuevo = pd.DataFrame({'const': 1, 'Nivel glucosa basal': [nivel_basal_nuevo]})
prediccion = modelo.predict(X_nuevo)

print(f"Estimación del nivel de glucosa a los 60 minutos para un paciente con nivel
```

```
                            OLS Regression Results
==============================================================================
Dep. Variable:     Nivel glucosa 60 min   R-squared:                       0.373
Model:                              OLS   Adj. R-squared:                  0.354
Method:                   Least Squares   F-statistic:                     19.61
Date:                  Mon, 04 Nov 2024   Prob (F-statistic):           9.83e-05
Time:                          15:15:17   Log-Likelihood:                -119.97
No. Observations:                    35   AIC:                             243.9
Df Residuals:                        33   BIC:                             247.0
Df Model:                             1
Covariance Type:              nonrobust
===========================================================================================
                              coef    std err          t      P>|t|      [0.025      0.9
75]
-------------------------------------------------------------------------------------------
---
const                   95.0928     12.004      7.921      0.000      70.669     119.
516
Nivel glucosa basal      0.6216      0.140      4.428      0.000       0.336       0.
907
==============================================================================
Omnibus:                        1.788   Durbin-Watson:                   1.841
Prob(Omnibus):                  0.409   Jarque-Bera (JB):                0.777
Skew:                          -0.108   Prob(JB):                        0.678
Kurtosis:                       3.698   Cond. No.                         791.
==============================================================================

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly spe
cified.
Estimación del nivel de glucosa a los 60 minutos para un paciente con nivel basal de
83 mg/dL: 146.68 mg/dL
```

In [17]:
```
# Obtener los coeficientes del modelo
intercepto = modelo.params['const']
pendiente = modelo.params['Nivel glucosa basal']

# Mostrar la fórmula de la regresión lineal
print(f"Ecuación de la regresión lineal: Y = {intercepto:.2f} + ({pendiente:.2f} *
```

```
Ecuación de la regresión lineal: Y = 95.09 + (0.62 * X)
```

El valor de la pendiente es de **0.6216**. Esto lo que nos indica es que por cada mg/Dl adicional en el nivel de glucosa basal, el nivel de glucosa pasados 60 min va a aumentar en **0.6216**.

Estimación para un paciente cuyo nivel basal es 83 mg/Dl:

Y = 95.09 + (0.62 * X) =

95.09+(0.621*83)= 146.6828**

Para un paciente cuyo nivel basal es de 83mg/Dl, se estima que pasados los 60 min su nivel basal es = **149.22 mg/Dl**

c) ¿Qué tanto por ciento del nivel de glucosa en sangre pasados 60 minutos queda no queda explicado por el anterior modelo?

Para poder saber que porcentaje no queda explicado por el modelo lineal entre el nivel de glucosa basal y el nivel de glucosa pasados 60 min, se necesita primero conocer el porcentaje explicado que en este caso es del seria 0.61 elevado al cuadrado: 0.3721, por siguiente el no explicado seria 1-0.3721

Porcentaje explicado : 37.21 % Porcentaje no explicado : 62.79 %

D) Si aumentásemos el nivel basal de un paciente en 5 mg/Dl ¿Qué variación experimentaría su nivel de glucosa al cabo de 60 minutos?

0.6216*5= 3.108 mg/Dl

Si aumentásemos el nivel basal de un paciente en 5 mg/Dl ,la variacion que experimentaria el nivel de glucosa al cabo de 60 min seria de = 3.108 m g/Dl

EJERCICIO 3

A) Se quiere estudiar si se puede admitir que el nivel medio de glucosa en sangre en el momento de la ingestión en los jóvenes es 88 mg/Dl. Obtener el intervalo de confianza al 95% y al 99% para el nivel medio de glucosa en sangre de los jóvenes y posteriormente contesta a la cuestión planteada con los resultados obtenidos o con un contraste de hipótesis

In [18]:
```
import statsmodels.api as sm


# Definir la variable independiente (Nivel glucosa basal) y dependiente (Nivel gluc
X = jovenes['Nivel glucosa basal']
y = jovenes['Nivel glucosa 60 min']

# Añadir una constante a X para incluir el intercepto en el modelo
```

```
X = sm.add_constant(X)

# Crear el modelo de regresión lineal y ajustarlo
modelo = sm.OLS(y, X).fit()

# Realizar la estimación para un paciente con nivel basal de 88 mg/dL
nivel_basal_nuevo = 88
X_nuevo = pd.DataFrame({'const': 1, 'Nivel glucosa basal': [nivel_basal_nuevo]})

# Intervalo de confianza al 95%
prediccion_95 = modelo.get_prediction(X_nuevo).summary_frame(alpha=0.05)
print("Intervalo de confianza al 95%:")
print(prediccion_95[['mean', 'mean_ci_lower', 'mean_ci_upper']])

# Intervalo de confianza al 99%
prediccion_99 = modelo.get_prediction(X_nuevo).summary_frame(alpha=0.01)
print("\nIntervalo de confianza al 99%:")
print(prediccion_99[['mean', 'mean_ci_lower', 'mean_ci_upper']])
```

```
Intervalo de confianza al 95%:
         mean  mean_ci_lower  mean_ci_upper
0  149.789767     147.017015     152.562519

Intervalo de confianza al 99%:
         mean  mean_ci_lower  mean_ci_upper
0  149.789767     146.064704     153.514831
```

In [19]:
```
import scipy.stats as st

GC1_GB = df['Nivel glucosa basal'][df['Grupo de control']==1].reset_index().drop('i

IC95 = st.t.interval(confidence=0.95, df=len(GC1_GB)-1, loc=np.mean(GC1_GB), scale=
print(f'El intervalo de confianza al 95% para el nivel medio de glucosa en sangre d
print()
IC99 = st.t.interval(confidence=0.99, df=len(GC1_GB)-1, loc=np.mean(GC1_GB), scale=
print(f'El intervalo de confianza al 99% para el nivel medio de glucosa en sangre d
```

```
El intervalo de confianza al 95% para el nivel medio de glucosa en sangre de los jóv
enes es: (array([81.806697]), array([88.25044586]))

El intervalo de confianza al 99% para el nivel medio de glucosa en sangre de los jóv
enes es: (array([80.70303677]), array([89.35410608]))
```

Al estudiar los resultados obtenidos en los intervalos de confianza al 95% y 99%, se podria admitir que el nivel medio de glucosa en la sangre en el momento de la ingestión en los jóvenes de 88mg/Dl dentro de ambos intervalos de confianza, tanto para al **95%** como para **99%** debido a que al estudiar los resultados obtenidos podemos ver que se encuentra dentro de los limites de ambos intervalos de confianza.

B) Obtener los intervalos de confianza al 95% para la diferencia de medias en el nivel basal de glucosa entre adultos y jovenes e interpreta los resultados. ¿Se puede concluir que el nivel basal de glucosa de los jóvenes y los adultos es el mismo con nivel de significación del 5%? .Suponiendo que se cumplen las condiciones iniciales teóricas para obtener los intervalos de confianza

In [20]:
```
GC1_GB = df['Nivel glucosa basal'][df['Grupo de control']==1].reset_index().drop('i
GC2_GB = df['Nivel glucosa basal'][df['Grupo de control']==2].reset_index().drop('i
n = 35
```

In [21]:
```
from scipy.stats import t

# Cálculo de la diferencia de medias
diff_mean = GC2_GB.mean()[0] - GC1_GB.mean()[0]

# Cálculo de las desviaciones estándar
sGC2_GB = GC2_GB.std(ddof=1)[0]
sGC1_GB = GC1_GB.std(ddof=1)[0]

# Cálculo del número de grados de libertad
dif = len(GC1_GB) + len(GC2_GB) - 2

# Cálculo del intervalo de confianza al 95%
ci_95 = t.interval(0.95, dif, loc=diff_mean, scale=((sGC2_GB**2 + sGC1_GB**2)/n)**0

# Cálculo del intervalo de confianza al 99%
ci_99 = t.interval(0.99, dif, loc=diff_mean, scale=((sGC2_GB**2 + sGC1_GB**2)/n)**0

print("Intervalo de confianza al 95%: ({:.2f}, {:.2f})".format(ci_95[0], ci_95[1]))
print("Intervalo de confianza al 99%: ({:.2f}, {:.2f})".format(ci_99[0], ci_99[1]))
```

```
Intervalo de confianza al 95%: (1.54, 10.07)
Intervalo de confianza al 99%: (0.13, 11.48)
```

```
C:\Users\josea\AppData\Local\Temp\ipykernel_15968\1695059526.py:4: FutureWarning: Se
ries.__getitem__ treating keys as positions is deprecated. In a future version, inte
ger keys will always be treated as labels (consistent with DataFrame behavior). To a
ccess a value by position, use `ser.iloc[pos]`
  diff_mean = GC2_GB.mean()[0] - GC1_GB.mean()[0]
C:\Users\josea\AppData\Local\Temp\ipykernel_15968\1695059526.py:7: FutureWarning: Se
ries.__getitem__ treating keys as positions is deprecated. In a future version, inte
ger keys will always be treated as labels (consistent with DataFrame behavior). To a
ccess a value by position, use `ser.iloc[pos]`
  sGC2_GB = GC2_GB.std(ddof=1)[0]
C:\Users\josea\AppData\Local\Temp\ipykernel_15968\1695059526.py:8: FutureWarning: Se
ries.__getitem__ treating keys as positions is deprecated. In a future version, inte
ger keys will always be treated as labels (consistent with DataFrame behavior). To a
ccess a value by position, use `ser.iloc[pos]`
  sGC1_GB = GC1_GB.std(ddof=1)[0]
```

Para determinar si se puede concluir que el nivel basal de glucosa es el mismo entre jóvenes y adultos, debemos observar si el intervalo de confianza al 95% incluye el valor 0:

Intervalo al 95%: (1.54, 10.07): Este intervalo **no incluye** el 0, lo que significa que existe evidencia suficiente para rechazar la hipótesis nula (que establece que no hay diferencia en los niveles de glucosa entre los grupos) al nivel de significación del 5%. Por lo tanto, se puede concluir que hay una diferencia significativa en los niveles basales de glucosa entre jóvenes y adultos.

C) Se quiere estudiar la proporción de la población con un nivel basal de glucosa superior a 95 mg/Dl (prediabetes). A partir de la muestra del fichero (tomando todos los datos) obtener un intervalo de confianza al 98% y contrastar la hipótesis que la proporción de la población con glucosa superior a 95 mg/Dl es 0,15 con nivel de significación del 5%.

In [22]:
```
import statsmodels.stats.proportion as smp
import scipy.stats as stats
GB = df['Nivel glucosa basal']
n = len(GB)
p = len(df[df['Nivel glucosa basal']> 95])
alpha = 0.02 # 1 - 0.98, ya que el nivel de confianza es del 98%

ci_98 = smp.proportion_confint(p, n, alpha=alpha, method='normal')

print("Intervalo de confianza al 98%: ({:.2f}, {:.2f})".format(ci_98[0], ci_98[1]))
```

```
Intervalo de confianza al 98%: (0.13, 0.39)
```

El valor de 0.15 mg/Dl de la proporcion de la poblacion con glucosa superior a 95, **se encuentra dentro del rango** del intervalor de confianza al 98%.

Se procede a hacer el contraste de hipotesis

In [23]:
```
alpha = 0.05
zstat, pvalue = smp.proportions_ztest(p, n, value=0.15, alternative='two-sided', pr

z_critico = stats.norm.ppf(alpha/2, n)

print('|z| = ',abs(zstat))
print('|z_critico| = ',abs(z_critico))


if abs(zstat) > abs(z_critico):
    print(f"Se rechaza la hipótesis nula al {(1-alpha)*100}% de confianza")
else:
    print(f"Se acepta la hipótesis nula al {(1-alpha)*100}% de confianza")
```

```
|z| =  2.0462071860002875
|z_critico| =  63.04003601545995
Se acepta la hipótesis nula al 95.0% de confianza
```

Si (Z) crítico > (Z): **No se rechaza la hipótesis nula** ((H_0)).

Esto nos da a entender que no hay evidencia suficiente para afirmar que hay una diferencia significativa entre los grupos o condiciones que se están comparando.

Al ser el valor Z (2.04) menor al valor de Z critico (63.04) se acepta la hipótesis nula al 95% de confianza y se puede afirmar que la proporción con glucosa basal es superior al 95 mg/dl de 0.15 de proporcion.