In [41]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
```

In [42]:
```
df=pd.read_csv('/content/StudentsPerformance.csv')
```

In [43]:
```
df.head()
```

Out[43]:

	gender	race/ethnicity	parental level of education	lunch	test preparation course	math score	reading score	writing score
0	female	group B	bachelor's degree	standard	none	72	72	74
1	female	group C	some college	standard	completed	69	90	88
2	female	group B	master's degree	standard	none	90	95	93
3	male	group A	associate's degree	free/reduced	none	47	57	44
4	male	group C	some college	standard	none	76	78	75

In [ ]:
```

```

In [44]:
```
df.isnull().sum()
```

Out[44]:

	0
gender	0
race/ethnicity	0
parental level of education	0
lunch	0
test preparation course	0
math score	0
reading score	0
writing score	0

dtype: int64

In [45]:

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1000 entries, 0 to 999
Data columns (total 8 columns):
 #   Column                       Non-Null Count  Dtype 
---  ------                       --------------  ----- 
 0   gender                       1000 non-null   object
 1   race/ethnicity               1000 non-null   object
 2   parental level of education  1000 non-null   object
 3   lunch                        1000 non-null   object
 4   test preparation course      1000 non-null   object
 5   math score                   1000 non-null   int64 
 6   reading score                1000 non-null   int64 
 7   writing score                1000 non-null   int64 
dtypes: int64(3), object(5)
memory usage: 62.6+ KB
```

In [46]:

```
df.describe()
```

Out[46]:

	math score	reading score	writing score
count	1000.00000	1000.000000	1000.000000
mean	66.08900	69.169000	68.054000
std	15.16308	14.600192	15.195657
min	0.00000	17.000000	10.000000
25%	57.00000	59.000000	57.750000
50%	66.00000	70.000000	69.000000
75%	77.00000	79.000000	79.000000
max	100.00000	100.000000	100.000000

EDA

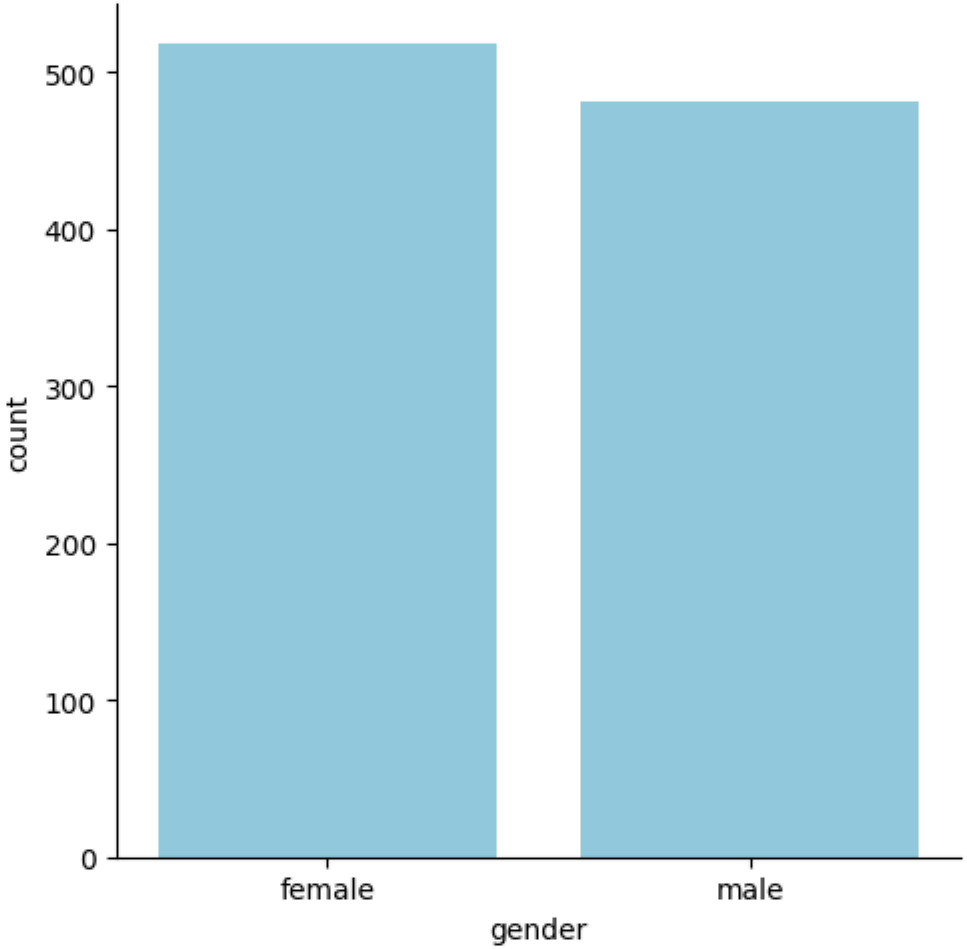
Univariate Analysis

In [47]:

```
sns.catplot(x='gender',data=df,kind='count',color='skyblue')
```

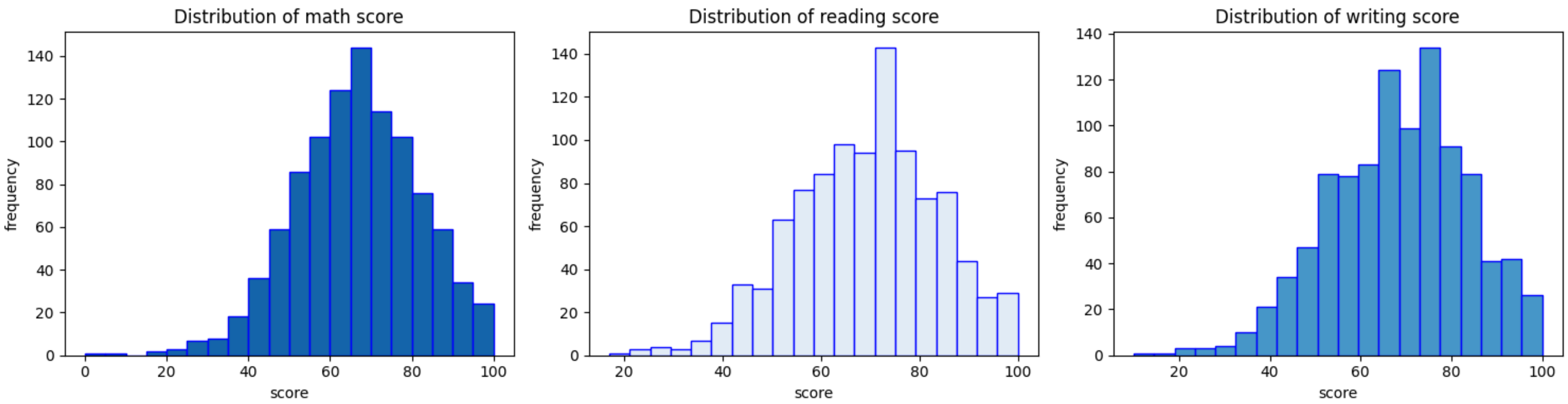
Out[47]:

```
<seaborn.axisgrid.FacetGrid at 0x7f7041820990>
```



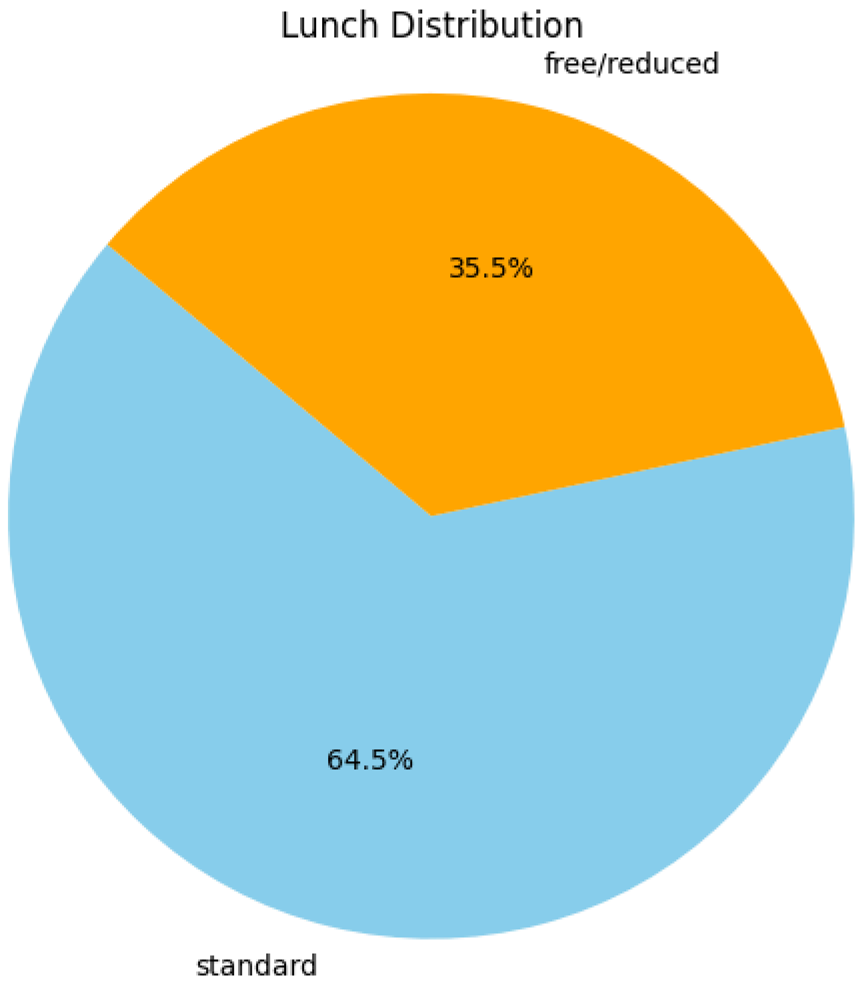
In [48]:

```
from matplotlib import cm
Score_columns = ['math score', 'reading score', 'writing score']
bins_color = [0.8, 0.1, 0.6]  # shades between 0 and 1
blue_colors = [cm.Blues(shade) for shade in bins_color]

plt.figure(figsize=(15, 4))
for i, col in enumerate(Score_columns):
    plt.subplot(1, 3, i+1)
    plt.hist(df[col], bins=20, color=blue_colors[i], edgecolor='blue')
    plt.xlabel('score')
    plt.ylabel('frequency')
    plt.title(f'Distribution of {col}')
    plt.tight_layout()
plt.show()
```



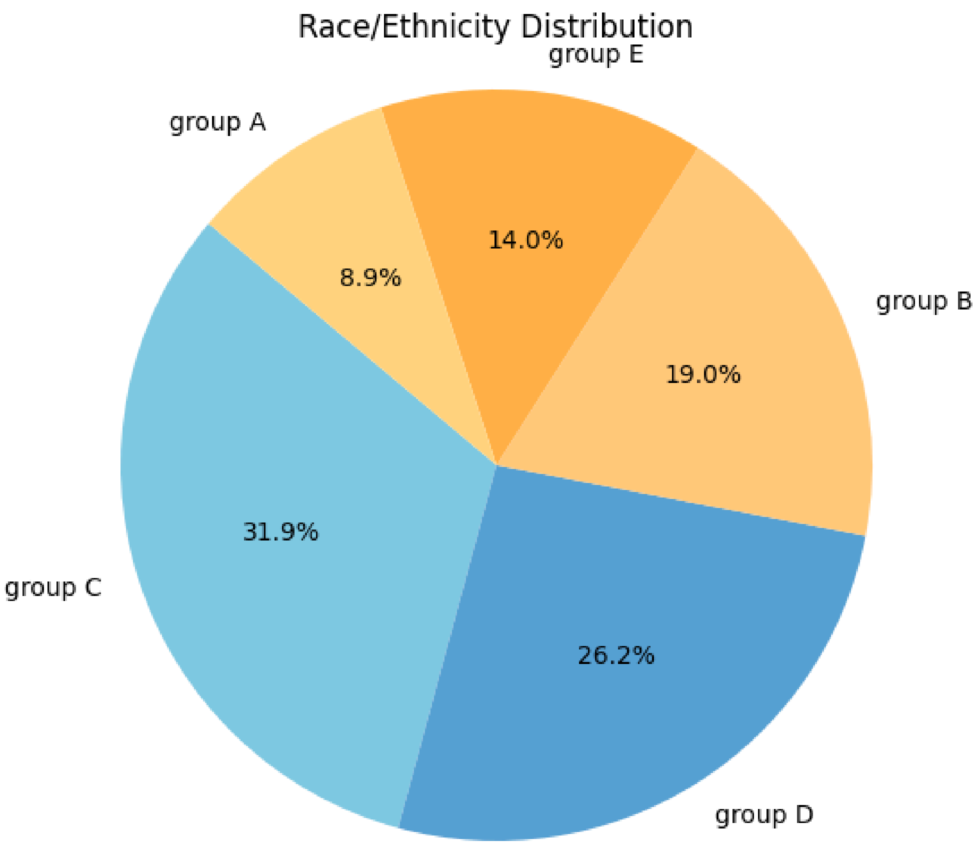
In [49]:

```
lunch_counts=df['lunch'].value_counts()
labels=lunch_counts.index
sizes=lunch_counts.values
colors = ['skyblue', 'orange']
plt.figure(figsize=(6, 6))

plt.pie(sizes,labels=labels,colors=colors,autopct='%1.1f%%',startangle=140)
plt.axis('equal')
plt.title('Lunch Distribution')
plt.show()
```

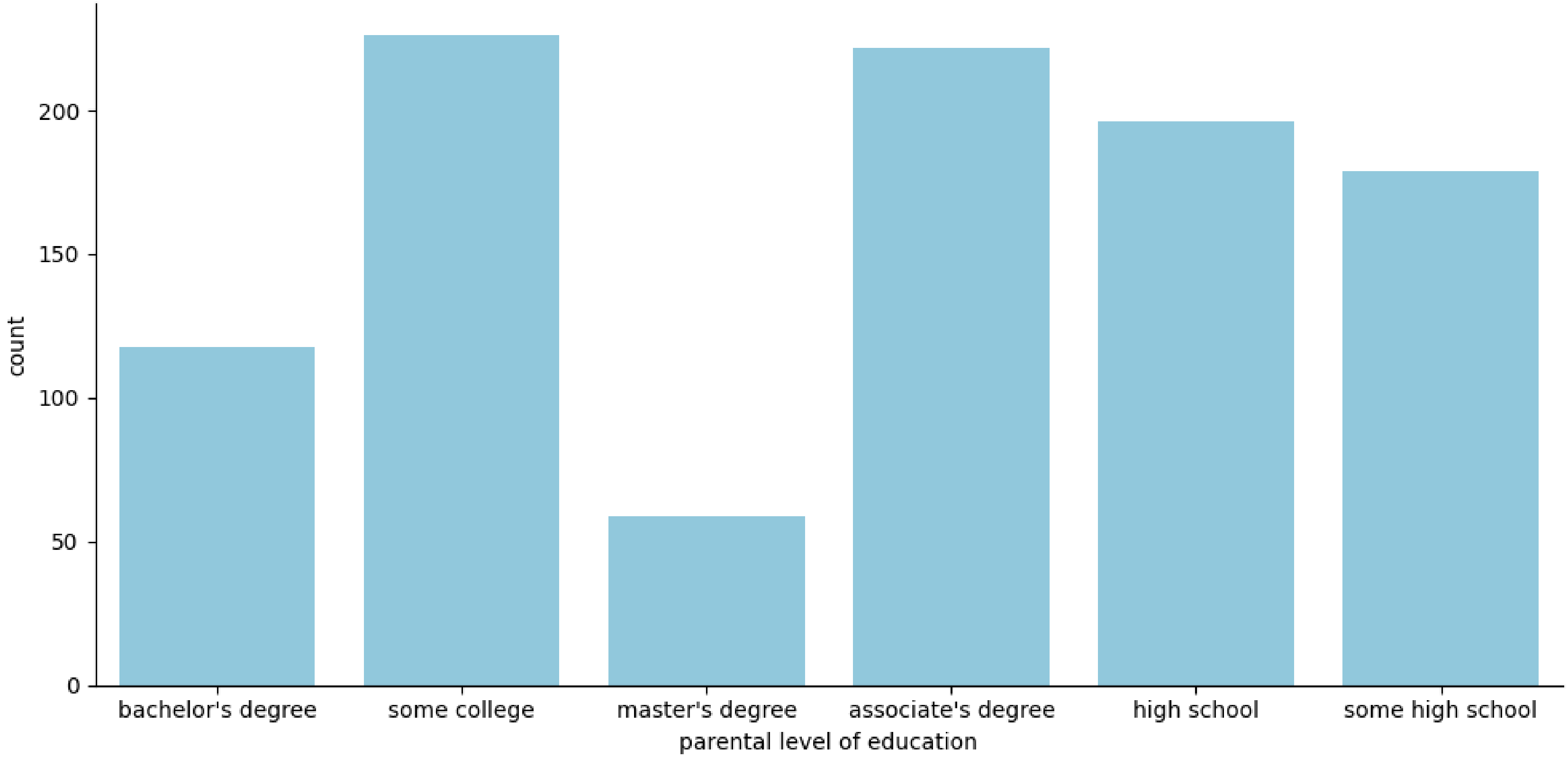


In [50]:

```
race_counts=df['race/ethnicity'].value_counts()
race_labels=race_counts.index
race_sizes=race_counts.values
race_colors = ['#7EC8E3', '#57A0D3', '#FFC87C', '#FFB347', '#FFD580']
plt.figure(figsize=(12, 6))
plt.pie(race_sizes, labels=race_labels, colors=race_colors,
        autopct='%1.1f%%', startangle=140)
plt.title('Race/Ethnicity Distribution')
plt.axis('equal')
plt.show()
```



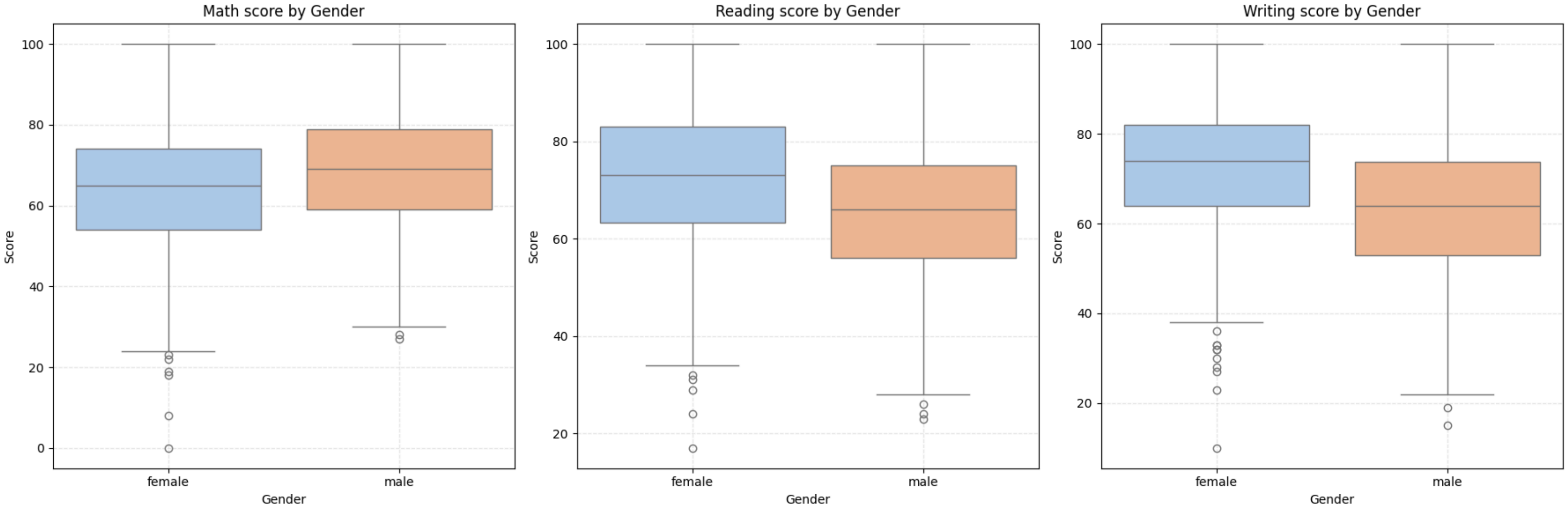
```
In [51]: sns.catplot(x='parental level of education',data=df,kind='count',color='skyblue',height=5,aspect=2)

Out[51]: <seaborn.axisgrid.FacetGrid at 0x7f703cd03050>
```



In [52]:

```
fig, axes = plt.subplots(1, 3, figsize=(18, 6))
score_columns = ['math score', 'reading score', 'writing score']

for i, score in enumerate(score_columns):
    sns.boxplot(x='gender', y=score, data=df, palette='pastel', ax=axes[i])
    axes[i].set_title(f'{score.capitalize()} by Gender')
    axes[i].set_xlabel('Gender')
    axes[i].set_ylabel('Score')
    axes[i].grid(True, linestyle='--', alpha=0.3)

plt.tight_layout()
plt.show()
```

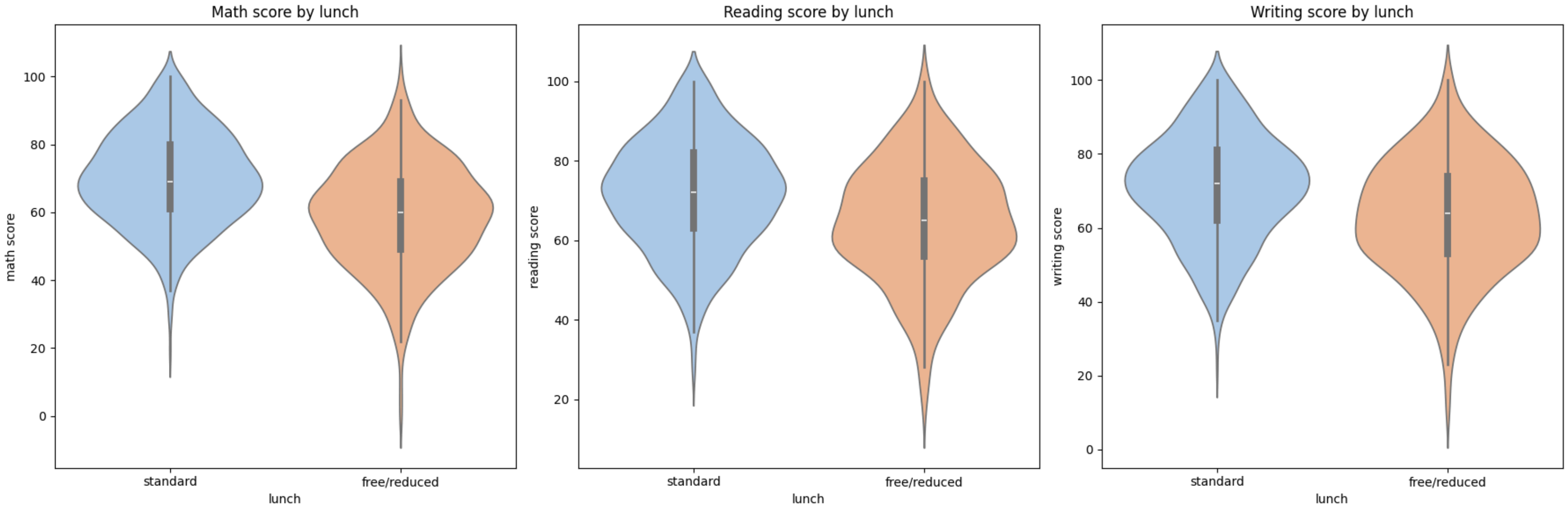
```
/tmp/ipython-input-3039254497.py:5: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `x` variable to `h
ue` and set `legend=False` for the same effect.

  sns.boxplot(x='gender', y=score, data=df, palette='pastel', ax=axes[i])
/tmp/ipython-input-3039254497.py:5: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `x` variable to `h
ue` and set `legend=False` for the same effect.

  sns.boxplot(x='gender', y=score, data=df, palette='pastel', ax=axes[i])
/tmp/ipython-input-3039254497.py:5: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `x` variable to `h
ue` and set `legend=False` for the same effect.

  sns.boxplot(x='gender', y=score, data=df, palette='pastel', ax=axes[i])
```



In [53]:
```
fig,axes=plt.subplots(1,3,figsize=(18,6))
for i, score in enumerate(score_columns):
  sns.violinplot(x='lunch',y=score,data=df,palette='pastel',ax=axes[i])
  axes[i].set_title(f'{score.capitalize()} by lunch')
  axes[i].set_xlabel('lunch')
```

```
plt.tight_layout()
plt.show()
```

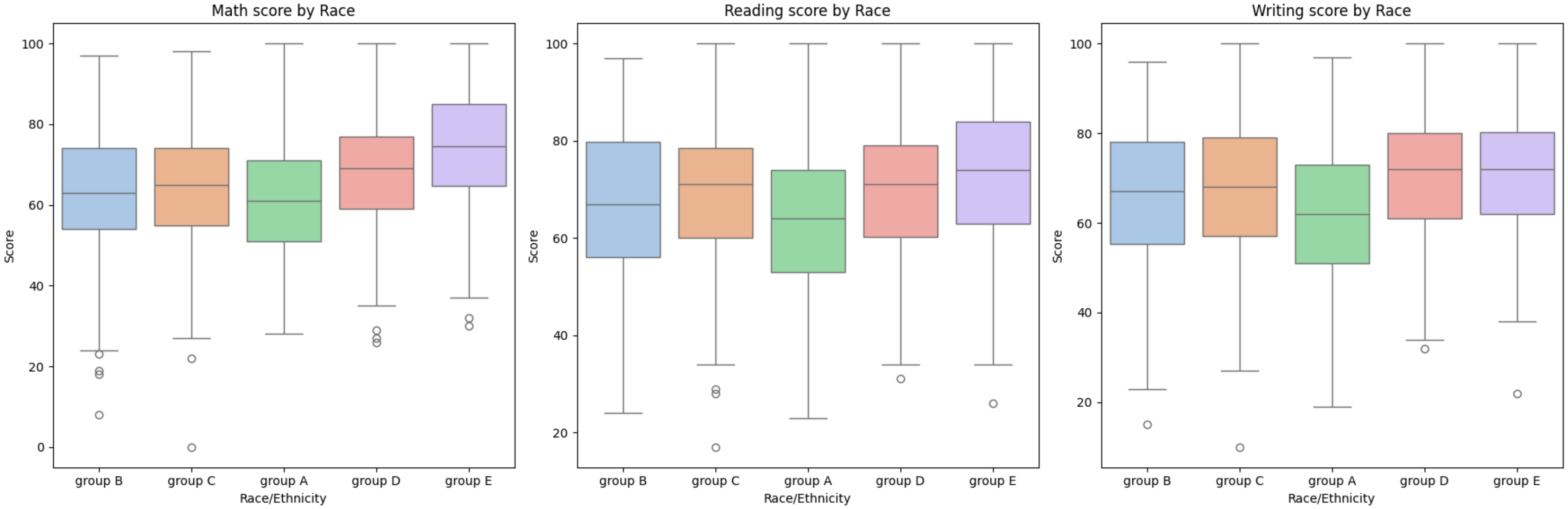
```
/tmp/ipython-input-2626086173.py:3: FutureWarning: 

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `x` variable to `h
ue` and set `legend=False` for the same effect.

  sns.violinplot(x='lunch',y=score,data=df,palette='pastel',ax=axes[i])
/tmp/ipython-input-2626086173.py:3: FutureWarning: 

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `x` variable to `h
ue` and set `legend=False` for the same effect.

  sns.violinplot(x='lunch',y=score,data=df,palette='pastel',ax=axes[i])
/tmp/ipython-input-2626086173.py:3: FutureWarning: 

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `x` variable to `h
ue` and set `legend=False` for the same effect.

  sns.violinplot(x='lunch',y=score,data=df,palette='pastel',ax=axes[i])
```



In [54]:

```
fig, axes = plt.subplots(1, 3, figsize=(18, 6))

for i, score in enumerate(score_columns):
    sns.boxplot(x='race/ethnicity', y=score, data=df, palette='pastel', ax=axes[i])
    axes[i].set_title(f'{score.capitalize()} by Race')
    axes[i].set_xlabel('Race/Ethnicity')
    axes[i].set_ylabel('Score')

plt.tight_layout()
plt.show()
```

```
/tmp/ipython-input-289488783.py:4: FutureWarning: 

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `x` variable to `h
ue` and set `legend=False` for the same effect.

  sns.boxplot(x='race/ethnicity', y=score, data=df, palette='pastel', ax=axes[i])
/tmp/ipython-input-289488783.py:4: FutureWarning: 

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `x` variable to `h
ue` and set `legend=False` for the same effect.

  sns.boxplot(x='race/ethnicity', y=score, data=df, palette='pastel', ax=axes[i])
/tmp/ipython-input-289488783.py:4: FutureWarning: 

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `x` variable to `h
ue` and set `legend=False` for the same effect.

  sns.boxplot(x='race/ethnicity', y=score, data=df, palette='pastel', ax=axes[i])
```


Math score by Race

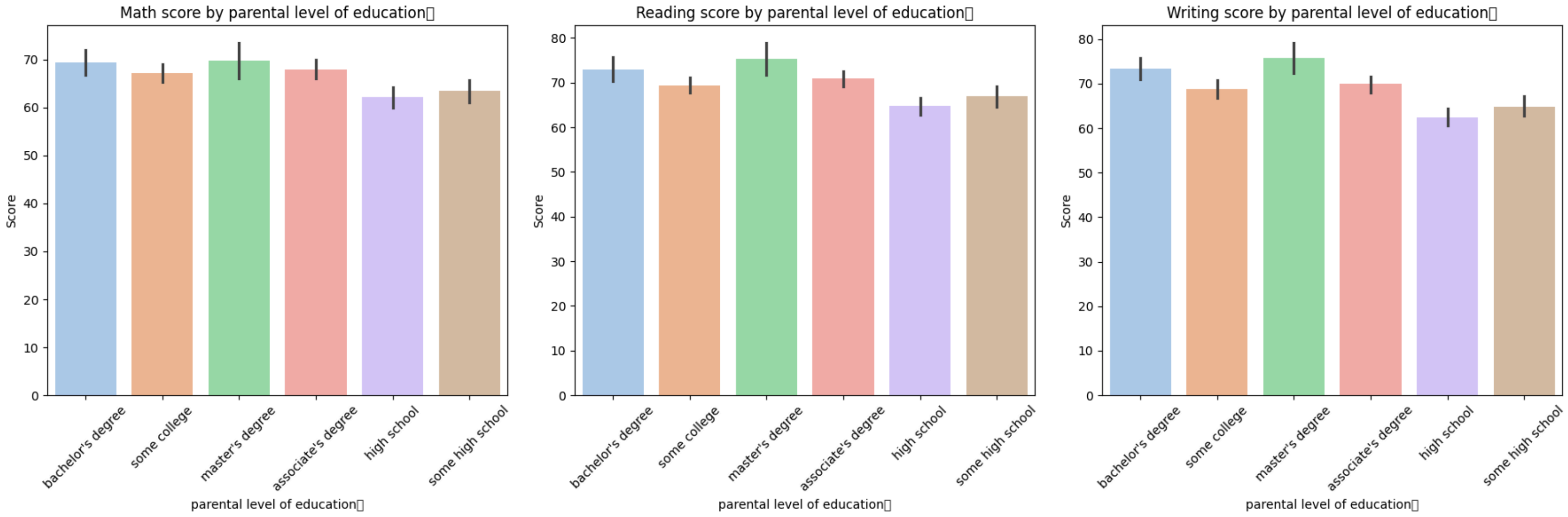
Reading score by Race

Writing score by Race

In [55]:

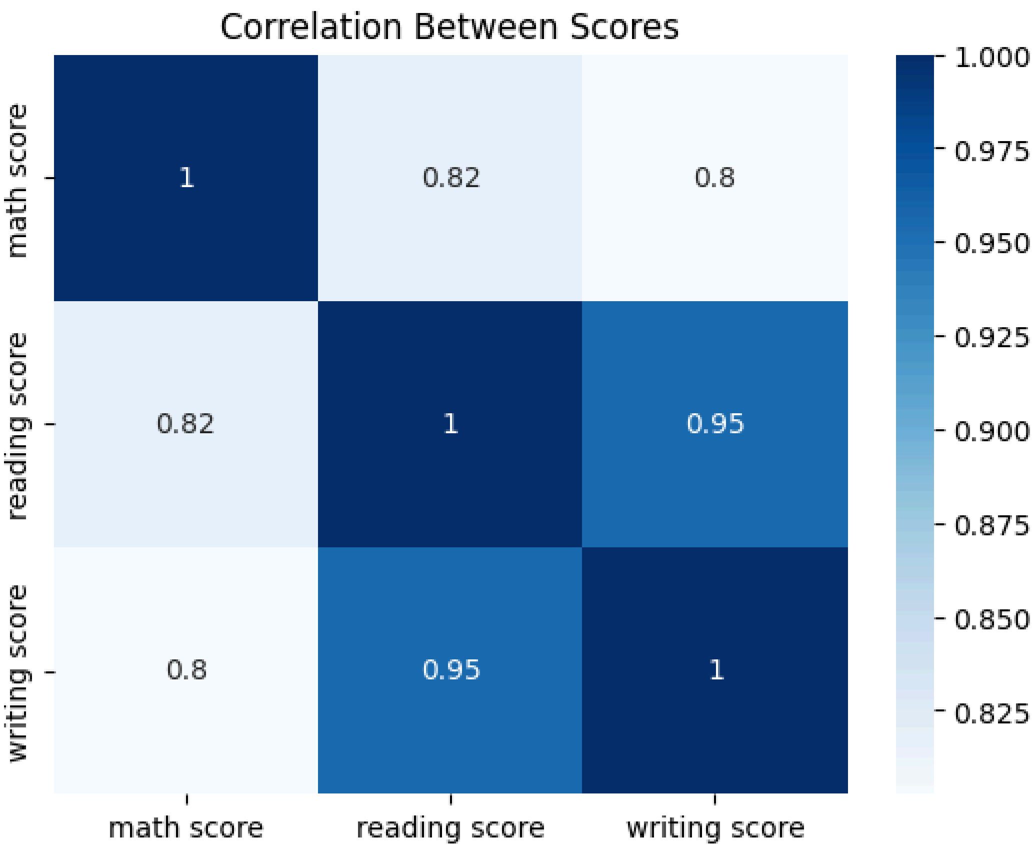
```
fig,axes=plt.subplots(1,3,figsize=(18,6))

for i, score in enumerate(score_columns):
  sns.barplot(x='parental level of education',y=score,data=df,palette='pastel',ax=axes[i])
  axes[i].set_title(f'{score.capitalize()} by parental level of education       ')
  axes[i].set_xlabel('parental level of education       ')
  axes[i].tick_params(axis='x', rotation=45)
  axes[i].set_ylabel('Score')
plt.tight_layout()
plt.show()
```

```
/tmp/ipython-input-806133456.py:4: FutureWarning: 

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `x` variable to `h
ue` and set `legend=False` for the same effect.

  sns.barplot(x='parental level of education',y=score,data=df,palette='pastel',ax=axes[i])
/tmp/ipython-input-806133456.py:4: FutureWarning: 

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `x` variable to `h
ue` and set `legend=False` for the same effect.

  sns.barplot(x='parental level of education',y=score,data=df,palette='pastel',ax=axes[i])
/tmp/ipython-input-806133456.py:4: FutureWarning: 

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `x` variable to `h
ue` and set `legend=False` for the same effect.

  sns.barplot(x='parental level of education',y=score,data=df,palette='pastel',ax=axes[i])
/tmp/ipython-input-806133456.py:9: UserWarning: Glyph 9 (	) missing from font(s) DejaVu Sans.
  plt.tight_layout()
/usr/local/lib/python3.11/dist-packages/IPython/core/pylabtools.py:151: UserWarning: Glyph 9 (	) missing from font
(s) DejaVu Sans.
  fig.canvas.print_figure(bytes_io, **kw)
```



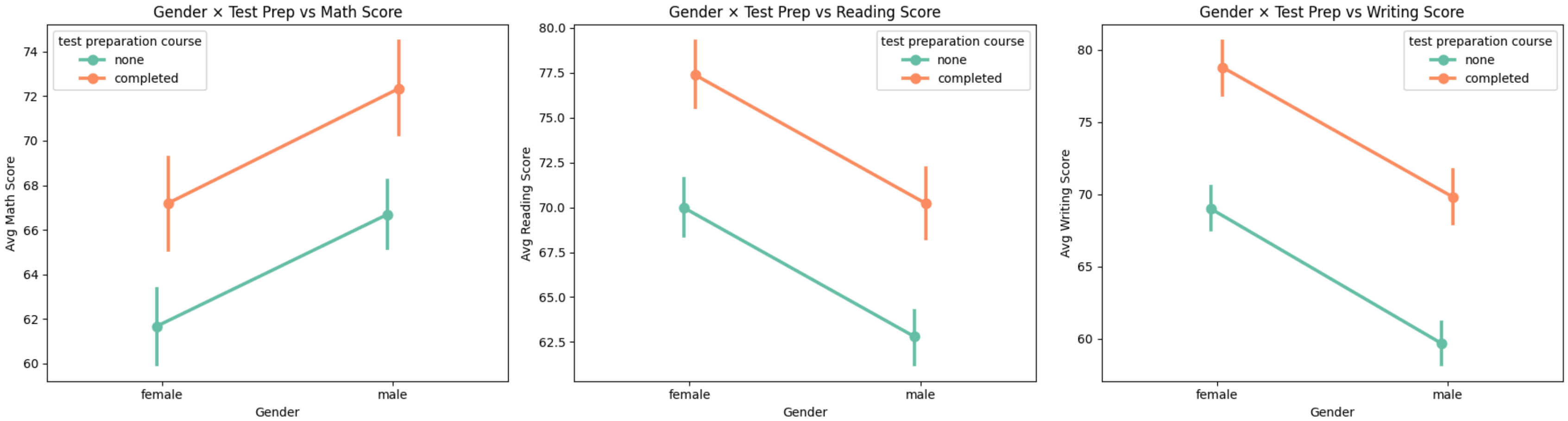
In [56]:

```
import seaborn as sns
import matplotlib.pyplot as plt
score_corr = df[['math score', 'reading score', 'writing score']].corr()
sns.heatmap(score_corr, annot=True, cmap='Blues')
plt.title('Correlation Between Scores')
plt.show()
```



In [57]:

```
score_columns = ['math score', 'reading score', 'writing score']

plt.figure(figsize=(18, 5))

for i, score in enumerate(score_columns):
    plt.subplot(1, 3, i + 1)
    sns.pointplot(data=df,
                  x='gender',
                  y=score,
                  hue='test preparation course',
                  palette='Set2',
                  dodge=True)
    plt.title(f'Gender × Test Prep vs {score.title()}')
    plt.ylabel(f'Avg {score.title()}')
    plt.xlabel('Gender')

plt.tight_layout()
plt.show()
```



In [65]:

```
from sklearn.linear_model import LinearRegression
from sklearn.multioutput import MultiOutputRegressor
from sklearn.model_selection import train_test_split
from sklearn.metrics import r2_score, mean_absolute_error, mean_squared_error
from sklearn.preprocessing import OneHotEncoder
from sklearn.compose import ColumnTransformer
from sklearn.pipeline import Pipeline
```

In [66]:

```
binary_cols = ['gender']
multiclass_cols = ['race/ethnicity', 'parental level of education', 'lunch', 'test preparation course']

X = df.drop(columns=['math score', 'reading score', 'writing score'])
y = df[['math score', 'reading score', 'writing score']]

preprocessor = ColumnTransformer(
    transformers=[
        ('bin', OneHotEncoder(drop='if_binary'), binary_cols),
        ('multi', OneHotEncoder(handle_unknown='ignore'), multiclass_cols)
    ],
    remainder='passthrough'
)
```

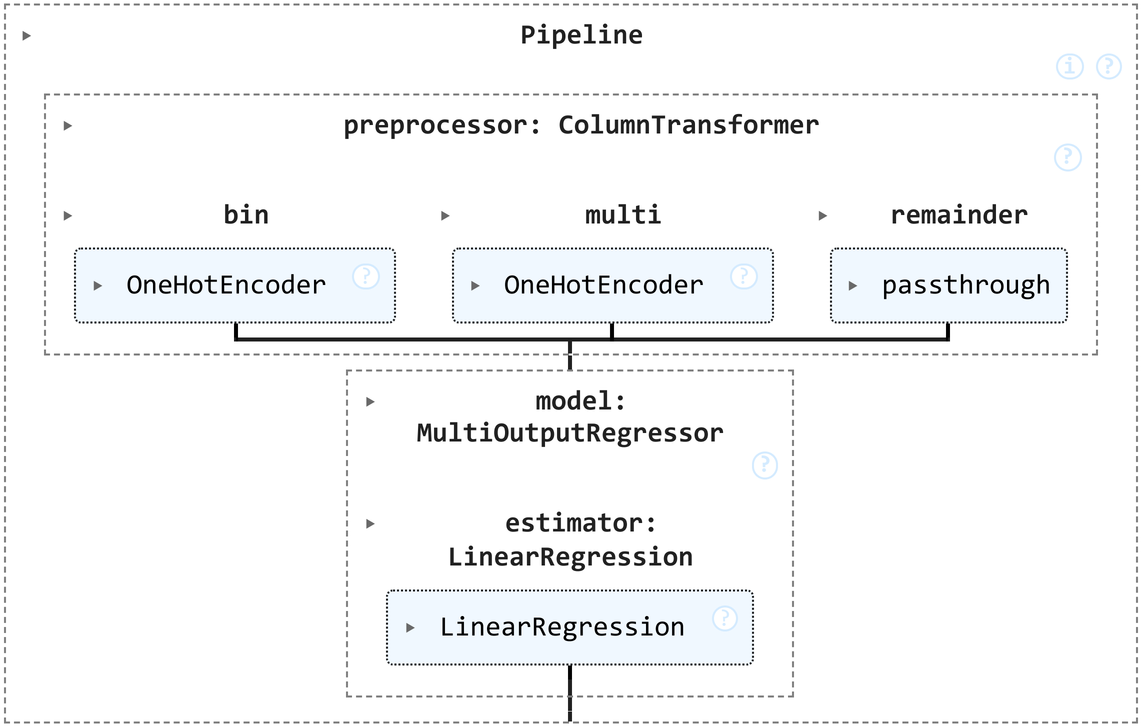
In [67]:

```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
```

In [68]:

```
pipeline = Pipeline([
    ('preprocessor', preprocessor),
    ('model', MultiOutputRegressor(LinearRegression()))
])

pipeline.fit(X_train, y_train)
```

Out[68]:



In [69]:

```
y_pred = pipeline.predict(X_test)
for i, target in enumerate(y.columns):
    print(f"Results for {target}:")
    print(f"R2 score: {r2_score(y_test.iloc[:, i], y_pred[:, i]):.3f}")
    print(f"MAE: {mean_absolute_error(y_test.iloc[:, i], y_pred[:, i]):.3f}")
    rmse = np.sqrt(mean_squared_error(y_test.iloc[:, i], y_pred[:, i]))
    print(f"RMSE: {rmse:.3f}\n")
```

```
Results for math score:
R2 score: 0.176
MAE: 11.270
RMSE: 14.160

Results for reading score:
R2 score: 0.159
MAE: 10.830
RMSE: 13.792

Results for writing score:
R2 score: 0.264
MAE: 10.193
RMSE: 13.321
```

In [70]:

```
preprocessor = pipeline.named_steps['preprocessor']
model = pipeline.named_steps['model']
def get_feature_names(column_transformer):
    feature_names = []
    for name, transformer, columns in column_transformer.transformers_:
        if name != 'remainder':
            if hasattr(transformer, 'get_feature_names_out'):
                names = transformer.get_feature_names_out(columns)
                feature_names.extend(names)
            else:
                feature_names.extend(columns)
        else:
            if transformer == 'passthrough':
                feature_names.extend(column_transformer._feature_names_in[column_transformer._feature_names_in.index(
    return feature_names

feature_names = get_feature_names(preprocessor)
for i, target in enumerate(y.columns):
    print(f"Coefficients for predicting {target}:")
    coefs = model.estimators_[i].coef_
    for feat, coef in zip(feature_names, coefs):
        print(f"  {feat}: {coef:.4f}")
    print()
```

```
Coefficients for predicting math score:
  gender_1: 4.5207
  race/ethnicity_0: -2.6951
  race/ethnicity_1: -2.5127
  race/ethnicity_2: -2.0922
  race/ethnicity_3: 0.9171
  race/ethnicity_4: 6.3829
  parental level of education_0: 0.4897
  parental level of education_1: 3.6088
  parental level of education_2: -3.6008
  parental level of education_3: 1.5643
  parental level of education_4: 0.3446
  parental level of education_5: -2.4067
  lunch_0: -5.7620
  lunch_1: 5.7620
  test preparation course_0: 2.9373
  test preparation course_1: -2.9373

Coefficients for predicting reading score:
  gender_1: -7.4137
  race/ethnicity_0: -1.4881
  race/ethnicity_1: -2.0185
  race/ethnicity_2: -0.8139
  race/ethnicity_3: 1.3239
  race/ethnicity_4: 2.9967
  parental level of education_0: 0.5641
  parental level of education_1: 3.6648
  parental level of education_2: -3.8098
  parental level of education_3: 2.4818
  parental level of education_4: -0.6786
  parental level of education_5: -2.2223
  lunch_0: -3.7328
  lunch_1: 3.7328
  test preparation course_0: 3.7856
  test preparation course_1: -3.7856

Coefficients for predicting writing score:
  gender_1: -9.3826
  race/ethnicity_0: -2.1283
  race/ethnicity_1: -2.1998
  race/ethnicity_2: -0.6516
  race/ethnicity_3: 2.7901
```

```
race/ethnicity_4: 2.1895
parental level of education_0: 0.6229
parental level of education_1: 4.9095
parental level of education_2: -4.8835
parental level of education_3: 3.4159
parental level of education_4: -0.5516
parental level of education_5: -3.5132
lunch_0: -4.3167
lunch_1: 4.3167
test preparation course_0: 5.0937
test preparation course_1: -5.0937
```