CSE 163 Syntax Reference Sheet

Python & data structures

Operators

```
+ - * / // % **
== != < > <= >=
and  or  not  in
```

Lists / Tuples

```
lst = []
lst.append(x)
lst[i]    lst[a:b]
len(lst)  sorted(lst, key=)
min(lst)  max(lst)
(a, b)
(a, b) = tup
```

Dictionaries

```
d = {}
d[key] = value
d.keys()  d.values()  d.items()
```

Sets

```
st = set()
st.add(x)
```

Strings

```
s.strip()   s.split(sep=)
s.lower()   s.upper()
sep.join(items)
f"{x} and {y}"
```

Loops / Conditionals

```
for item in seq:
    ...
for i, x in enumerate(seq):
for a, b in zip(s1, s2):
range(n)  range(a, b)  range(a, b, step)
if cond:
elif cond:
else:
```

Files

```
with open(path) as f:
f.read()
f.readlines()
```

Pandas (pd)

```
pd.read_csv(path)
df.head()   df.shape   df.columns

df["col"]
df[["c1", "c2"]]
df.loc[rows, cols]
df.iloc[i, j]
df[(df["a"] == x) & (df["b"] > y)]
df["col"].str.upper()
df["col"].apply(my_func)

df["col"].unique()   df.notnull()
df["col"].isnull()   df.dropna()

df["new_col"] = values
df["col"].fillna(value)
df["col"].astype(dtype)

df.set_index("c1")
df.set_index(["c1", "c2"])
df.reset_index()

df.groupby("c")["c2"].mean()
df.groupby(["c1","c2"])["c3"].sum()
#  .mean() .sum() .count()
#  .max()  .min()  .median()

s.max()  s.min()  s.mean()
s.idxmax()   s.idxmin()
df.nlargest(n, "col")

df.merge(other, on=,
   left_on=, right_on=,
   how="inner"|"outer"|"left"|"right")

pd.get_dummies(df)
```

Plotting (plt, sns)

```
fig, ax = plt.subplots(nrows, ncols, figsize=)

sns.relplot(data=, x=, y=,
   hue=, kind="line"|"scatter")
sns.catplot(data=, x=, y=,
   hue=, kind="bar"|"count")
sns.histplot(data=, x=, ax=)

plt.title()
plt.xlabel()   plt.ylabel()
plt.legend()
plt.savefig(path)
```

GeoPandas (gpd)

```
gpd.read_file(path)

gpd.sjoin(left, right, how=)
gdf.dissolve(by=, aggfunc=)
gdf.merge(other, on=)

gdf.plot(column=, legend=,
   ax=, color=)
```

scikit-learn

```
train_test_split(X, y, test_size=)

DecisionTreeClassifier(max_depth=)
DecisionTreeRegressor(max_depth=)

model.fit(X_train, y_train)
model.predict(X_test)

accuracy_score(y_true, y_pred)
mean_squared_error(y_true, y_pred)
```