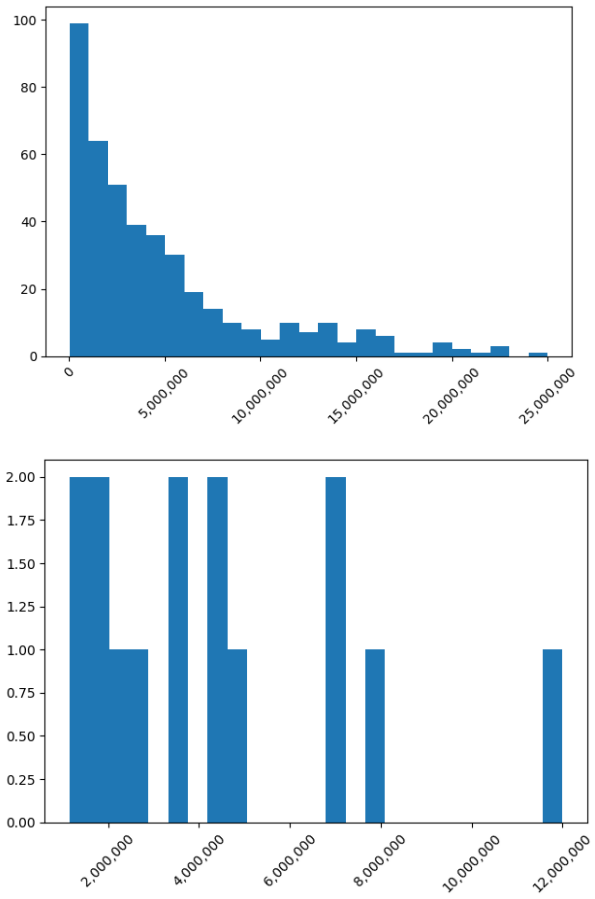
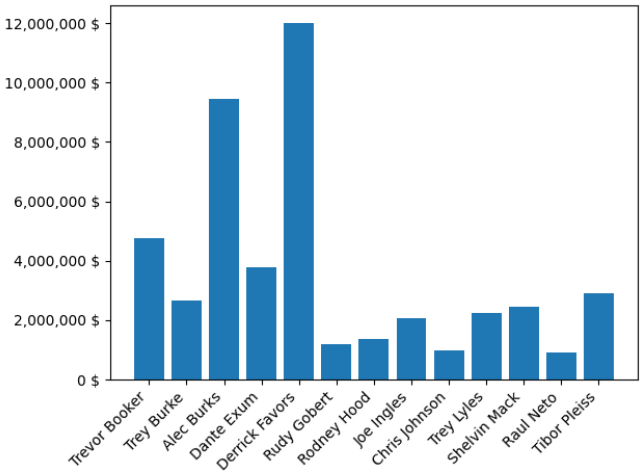
Temat: Pandas – Analiza danych, część 2

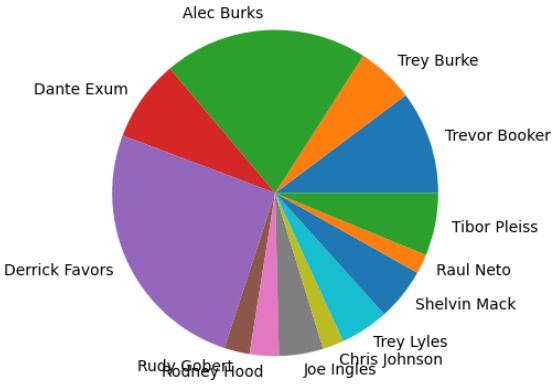
Wykonaj zadanie zgodnie z poniższymi wymaganiami:

1. Zaimportuj bibliotekę ***pandas*** i nazwij ją jako ***pd***.
2. Zaimportuj moduł ***pyplot*** z biblioteki ***matplotlib***
3. Zaimportuj dane z plików: *players-data.csv, players-collage.csv, players-salary.csv*. Wczytane dane umieść odpowiednio w zmiennych: ***players***, ***collages***, ***salaries***
4. Wyświetl po 3 pierwsze rekordy zaimportowanych danych oraz wyświetl informacje o kolumnach.
5. Połącz wszystkie tabele (***players***, ***collages*** *i* ***salaries***) tak aby w wyniku pojawiły się tylko dane istniejące. Wynik połączenia umieść w zmiennej ***merged_inner***.
6. Wyświetl połączone dane (***merged_inner***), oraz informacje o kolumnach. Sprawdź ilość rekordów w połączonej tabeli.
7. Połącz tabele ***players***, ***collages*** *i* ***salaries***, tabela ***players*** jest nadrzędna. Wyniki połączenia umieść w zmiennej ***merged_left***.
8. Wyświetl połączone dane (***merged_left***), oraz informacje o kolumnach. Sprawdź ilość rekordów w połączonej tabeli.
9. Skonwertuj wzrost i wagę na jednostki metryczne (***cm i kg***).
10. Oblicz średnią wartość zarobków dla gracza w danym zespole. Użyj grupowania i funkcji agregacyjnej ***mean.*** Wynik grupowania umieść w zmiennej ***grp1***.
11. Sformatuj wartość średnich zarobków jako liczby zmiennoprzecinkowe: ***123456789.00***. Uwaga: zmiana formatowania, konwertuje typ ***float*** na ***object***!
12. Połącz tabele ***merged_left*** i tabelę ***grp1*** - wynik łączenia umieść w ***merged_left***
13. Wyświetl połączone dane, zwróć uwagę na typy danych kolumn!
14. Uzupełnij brakujące wynagrodzenia (NaN/NULL) w tabeli ***merged_left*** o obliczone średnie i zapisz w zmiennej ***merged_all.***
15. Usuń zbędną kolumnę (średnie zarobki dla drużyny)
16. Dodaj nową kolumnę zawierającą zarobki w formacie ***123 456 789.00 $*** i nazwij ją ***SalaryUSD.***
17. Wyświetl dane oraz informacje o kolumnach (zwróć uwagę na typy danych).
18. Graficzna reprezentacja danych (Histogram)
    a. Wyświetl wykres histogramu przedstawiający zarobki wszystkich graczy ***Salary*** (podziel wykres na 25 zakresów).
    b. Przygotuj nowy obiekt, zawierający tylko dane dla drużyny ***Boston Celtics***. Wyświetl te dane.
    c. Wyświetl wykres histogramu przedstawiający zarobki graczy ***Salary*** dla tej drużyny (podziel wykres na 25 zakresów).
    d. Sformatuj obszary wykresu w taki sposób, aby zarobki były w formacie: ***123 456 789.00 $.***



19. Graficzna reprezentacja danych (wykres słupkowy i kołowy):
    a. Przygotuj nowy obiekt, zawierający tylko dane dla drużyny ***Utah Jazz***. Wyświetl te dane.
    b. Wyświetl wykres słupkowy przedstawiający zarobki tych graczy ***Salary***
    c. Sformatuj obszary wykresu w taki sposób, aby zarobki były w formacie: ***123 456 789.00 $.***
    d. Wyświetl wykres kołowy przedstawiający procentowy udział zarobków graczy dla tej drużyny





Dokumentacja.

Funkcje i metody, które mogą być konieczne do wykonania zadania:

1. ***len(obiekt)*** – zwraca ilość elementu obiektu (stringu, listy, innych zaawansowanych obiektów iterowalnych)
2. ***read_csv*** – zwraca obiekt DataFrame z danymi wczytanymi z pliku csv. Ścieżka do pliku jest argumentem obowiązkowym. Argument o nazwie ***sep*** umożliwia podanie innego niż standardowy separator pliku CSV (domyślnym separatorem jest przecinek).
3. ***head() i tail()*** – metody zwracające początkową i końcową część obiektu Series/DataFrame.
4. ***mean(), min(), max()*** – metody zwracające odpowiednio średnią, minimalną i maksymalną wartość liczbową, z kolumny (lub kolumn). Jeżeli kolumny posiadają wartości nie będące wartościami liczbowymi, metoda zwróci błąd.
5. ***loc[], iloc[]*** – atrybuty zwracające wiersz zgodnie z kryteriami. Odwołanie przez atrybuty ***loc*** i ***iloc*** zapewniają bezpieczny dostęp do danych niezależnie od zmiany indeksów (przeindeksowania DataFrame w trakcie filtrowania danych).

Filtrowanie danych obiektu DataFrame.

Przykładowe dane:

Id	A	B	C
1	23	PL	3.6
2	32	UK	4.5
3	23	UK	3.2
4	25	PL	4.6
5	12	US	21
6	33	US	3.6

```
# Wyświetlanie obiektu DataFrame o nazwie df, np. zaimportowanego z pliku CSV
print(df)
# Wyświetlanie tylko jednej kolumny np.:
print(df[["A"]])
# lub
print(df["A"])
# Wyświetlanie kilku kolumn:
print(df[["A","B"]])
# Wyświetlanie tylko kolumny A dla wierszy, które spełniają określone kryteria np. kolumna B ==
# US i kolumna C < 5
print(df["A"][df["B"]=="US"][df["C"]<5])
```

```
#lub bezpieczniej:
print(df["A"].loc[df["B"]=="US"].loc[df["C"]<5])
# Obliczanie średniej dla kolumny C, gdy wiersze z kolumny B są równe PL
print(df[df["B"]=="PL"]["C"].mean())
#lub bezpieczniej:
print(df.loc[df["B"]=="PL"]["C"].mean())
```

Informacje o typach danych tabeli:

```
Print(Obiekt_dataframe.info())
```

Łączenie danych z kilku tabel:

```
rodzaj_złączenia: ”left”,”right”,”inner”, (”outer”, ”cross”)
Wynik = pd.merge(left = lewa_tabela, right = prawa_tabela, how = rodzaj_złączenia)
Wynik = lewa_tabela.merge(prawa_tabela, how = rodzaj_złączenia)
```

Konwertowanie i formatowanie danych

#konwersja typu na float (jeżeli to możliwe)

```
tabela["kolumna"] = tabela["kolumna"].map(lambda x: float(x)) #funkcja anonimowa, tzw. lambda function
#formatowanie np. pola walutowego 1,000,000.00 $:
tabela["nowa_kolumna"] = tabela ["stara_kolumna"].map('{:,.2f} $'.format)
```

Uzupełnianie danych

```
tabela['kolumna_docelowa’][ tabela ['kolumna_docelowa'].isna()] = tabela["kolumna_źródłowa"]
```

Wykres histogramu

```
plt.hist(tabela["kolumna_z_liczbami"], bins = ilość_zakresów)
plt.xticks(rotation=45)  #rotacja etykiet
#konwersja etykiet z liczb na tekst w określonym formacie np. 1000,000.00
xlabels = plt.gca().get_xticks().tolist()
plt.gca().set_xticklabels(['{:,.0f}'.format(x) for x in xlabels])
plt.tight_layout() #dopasowanie obszaru wykresu do zawartości
plt.show()
```

Wykres słupkowy

```
plt.bar(tabela["kolumna_z_etykietami"],tabela["kolumna_z_wartościami"])
plt.show()
```

Wykres kołowy

```
plt. pie(tabela["kolumna_z_wartościami"],labels= tabela["kolumna_z_etykietami"])
plt.show()
```