Projekt

NBA, izvozi podatke za različne igralce in različne ekipe

1. Csv datoteka
2. Jupyter file
3. Github file
4. Pripravi vhodne tabele, glede na različne igralce, ekipe,…
5. Analiza podatkov:

* razvrstitev igralcev glede na število točk, starost, rebounds performance,… , morda pokaži odvisnost števila točk/performance glede na starost
* razvrstitev ekip glede na povprečno število točk, razvrsti ekipe glede na to, koliko zmag imajo proti neki ekipi (npr. ekipe razvrstimo glede na to, katera ima več zmag proti Nuggetsom)

1. Predstavi podatke z grafi, npr., kako se je število točk nekega igralca spreminjalo skozi sezono
2. Poglej še kakšne funkcije, ki jih nismo delali na predavanjih in bi izboljšale program

Točkovnik

**Analiza podatkov: 20 točk**

* Vseh **20 točk** dobite, če napišete programe, ki zajamejo izbrane podatke iz interneta, jih ustrezno predelajo, shranijo, analizirajo in predstavijo rezultate analize.
* Za vsak manjkajoči del se ustrezno odbije nekaj točk .
* Če napišete samo funkcije, ki jih mora uporabnik poganjati iz Pythona, projektne naloge **ne morete opraviti**.

**Zahtevnost programa: 30 točk**

Tu **ni natančnega kriterija**, zato raje vprašajte za mnenje. Zahtevnejše kot je pridobivanje in analiza podatkov več točk dobite.

**Uporabniška izkušnja: 20 točk**

Tudi tu **ni natančnega kriterija**. Nekaj stvari, na katere morate biti pozorni:

* Rezultate predstavite z Jupyter Notebook.
* Rezultati naj bodo smiselno urejeni in opisani. Zaželene so tudi grafične predstavitve.
* V besedilu pazite na slovnične napake.

**Urejenost kode: 20 točk**

* Za skladnost s [PEP 8](https://www.python.org/dev/peps/pep-0008/), torej da ustrezno uporabljate presledke, nimate predolgih vrstic in podobno, lahko dobite **do 5 točk**.
* Če je vaš program zasnovan tako, so vsebinsko ločene funkionalnosti tudi ustrezno organizirane, dobite **do 5 točk**.
* Za elegantno kodo lahko dobite **do 10 točk**. Tu gre za to, da ponavljajoče ukaze prenesete v funkcije ali zanke, da definicije funkcij niso predolge (sicer jih razbijte na manjše), da ustrezno poimenujete spremenljivke, da kodo pišete konsistentno, uporabljate primerne podatkovne strukture (kdaj se uporabi množico, kdaj seznam, kdaj nabor, …) in podobno.

**Urejenost repozitorija: 10 točk**

* Za lepo napisan README.md, ki vsebuje kratek opis programa ter navodila za uporabo, lahko dobite **do 5 točk**. Če je vaš program zasnovan tako, da že vsebuje navodila za uporabo, je dovolj, da README.md vsebuje samo navodila za zagon.
* Če v repozitoriju ni odvečnih stvari kot mape \_\_pycache\_\_, datotek \*.pyc, starih datotek (projektna-prejsnja.py, poskus.py, …) in nič odvečnih datotek, dobite **do 3 točke**. Najboljši način, da se izognete odvečnim datotekam, je seveda uporaba .gitignore.
* Če so sporočila v spremembah jasna (in ne le delo ali celo lksdjf), dobite **do 2 točki**. Če spremembe delate prek GitHuba (in so sporočila Uploaded via GitHub in podobno), točk ne boste dobili.