# Strukture podataka i algoritmi

## Projektni zadatak 002\_POKEMON

Vaš zadatak je napisati program (konzolnu aplikaciju u C++) koji će korisniku omogućiti rad s Pokemonima. Program mora biti smisleno strukturiran i intuitivan. U nastavku su opisane funkcionalnosti koje treba program imati. Za izradu projekta se koristi datoteka **SPA\_PROJ\_002\_POKEMON\_data.csv** čiji je opis sljedeći (ako neki podatak nedostaje, zamijenite ga nulom ili praznim stringom):

* #: ID for each pokemon
* Name: Name of each pokemon
* Type 1: Each pokemon has a type, this determines weakness/resistance to attacks
* Type 2: Some pokemon are of dual type (Type 1 and Type 2). Type 1 is obligatory, and Type 2 is optional
* Total: sum of all stats that come after this, a general guide to how strong a pokemon is
* HP: hit points, or health, defines how much damage a pokemon can withstand before fainting
* Attack: the base modifier for normal attacks (eg. Scratch, Punch)
* Defense: the base damage resistance against normal attacks
* SP Atk: special attack, the base modifier for special attacks (e.g. fire blast, bubble beam)
* SP Def: the base damage resistance against special attacks
* Speed: determines which pokemon attacks first each round
* Generation: number of generation
* Legendary: is pokemon legendary or not

### Ishod 1 (bodova: 20)

1. (Bodova: **1**) Ispišite naslov programa koristeći ASCII art, primjerice, možete uzeti nešto s <https://www.asciiart.eu/> (ova funkcionalnost se svodi na jednostavan ispis red po red). Moguć izgled:

,'\

\_.----. \_\_\_\_ ,' \_\ \_\_\_ \_\_\_ \_\_\_\_

\_,-' `. | | /`. \,-' | \ / | | \ |`.

\ \_\_ \ '-. | / `. \_\_\_ | \/ | '-. \ | |

\. \ \ | \_\_ | |/ ,','\_ `. | | \_\_ | \| |

\ \/ /,' \_`.| ,' / / / / | ,' \_`.| | |

\ ,-'/ / \ ,' | \/ / ,`.| / / \ | |

\ \ | \\_/ | `-. \ `' /| | || \\_/ | |\ |

\ \ \ / `-.`.\_\_\_,-' | |\ /| \ / | | |

\ \ `.\_\_,'| |`-.\_ `| |\_\_| \/ | `.\_\_,'| | | |

\\_.-' |\_\_| `-.\_ | '-.| '-.| | |

`' '-.\_|

1. (Bodova: **2**) Dizajnirajte i implementirajte tip podataka Pokemon koji može čuvati podatke o pokemonima iz datoteke. Pripremite tip podataka tako da se može konstruirati na smislene načine te definirajte gettere, settere i potrebne metode za svakog člana. Omogućite korisniku da pozivanjem metode na objektu može ispisati osnovne podatke o pokemonu na ekran (naziv, tip 1, tip 2, total).
2. (Bodova: **2**) Prikažite korisniku opciju u izborniku „Unos putanje“. Kad korisnik odabere opciju, pitajte ga da upiše putanju foldera u kojemu se nalazi datoteka pa upisanu putanju spremite u binarnu datoteku.
3. (Bodova: **6**) Prikažite korisniku opciju u izborniku „Unos podataka iz datoteke“. Kad korisnik odabere opciju, otvorite datoteku (putanju foldera pročitajte iz binarne datoteke) te sav sadržaj iz nje isparsirajte u vektor Pokemona. Taj vektor sad predstavlja vašu bazu podataka pokemona i osnova je za sve ostale zadatke.
4. (Bodova: **2**) Izmjerite i ispišite koliko je trajalo učitavanje datoteke u vektor.
5. (Bodova: **7**) Promijenite tip podataka Pokemon tako da mu dodate novu metodu fight koja prima instancu drugog pokemona (napadača) i simulira njihovu borbu. U borbi prvo napada napadač i skida određenu količinu HP-a branitelju, a zatim branitelj uzvraća. Borba traje sve dok jedan od pokemona ne ostane bez HP-a, čime je drugi pokemon pobjednik. Za izračun količine HP-a koju treba skinuti u svakom napadu možete koristiti pojednostavljenu formulu:

Alternativno, umjesto ove formule možete koristiti neku koja bolje odgovara stvarnoj borbi pokemona (primjerice, <https://www.math.miami.edu/~jam/azure/compendium/battdam.htm>).

### Ishod 2 (bodova: 20)

1. (Bodova: **3**) Prikažite korisniku opciju u izborniku „Kopiranje datoteke“. Kad korisnik odabere opciju, pitajte ga želi li prepisati retke iz datoteke u istom ili obrnutom redoslijedu. Prikladno koristeći stog ili red, prepišite retke u željenom redoslijedu iz originalne datoteke u novu datoteku u istom folderu.
2. (Bodova: **3**) Prikažite korisniku opciju u izborniku „Obrada otrovnih pokemona“. Kad korisnik odabere opciju, napravite sljedeće operacije:

* Učitajte sve „Poison“ pokemone u novu listu (koristite list<T>).
* Nakon učitavanja, obrišite sve pokemone iz liste čiji je „Total“ manji od 300.
* Preostalim pokemonima pomnožite HP sa 1.5
* Poslažite preostale pokemone u listi obrnutim redoslijedom (prvi postaje zadnji, drugi predzadnji, …)
* Ispišite sve pokemone iz liste.

1. (Bodova: **4**) Simulirajte dolazak pokemona na stadion. Pomiješajte sve pokemone u vektoru slučajnim redoslijedom i prepišite ih u vektor (ispišite „Pokemon *xy* je stao u red za karte“). Nakon toga, izvadite sve pokemone iz reda (ispišite „Pokemon *xy* je usao na stadion“).
2. (Bodova: **5**) Prikažite korisniku opciju u izborniku „Histogram“. Kad korisnik odabere opciju, obradite sve pokemone iz vektora na način da iscrtate histogram prema brzini (svojstvo „Speed“). Definirajte sljedeće intervale:   
   0-19, 20-39, 40-59, 60-79, 80-99, 100+. Ispišite ukupan broj pokemona u svakom intervalu te ispišite potreban broj znakova „#“ (svaki znak predstavlja najviše 10 pokemona u tom intervalu). Primjer prikaza (intervali i brojevi su izmišljeni):

0-49: ### (25)

50-100: # (1)

100+: ##### (49)

1. (Bodova: **5**) Korištenjem pomoćnog stoga, simulirajte transakcije na vašem vektoru koji predstavlja bazu podataka. Postupite ovako:

* Omogućite korisniku da zadaje naredbe.
* Kad korisnik zada naredbu „BEGIN TRAN“, transakcija počinje.
* Sad korisnik može zadavati naredbu „ADD HP *n*“ koliko puta želi. Svakim zadavanjem naredbe svim pokemonima povećajte svojstvo HP za iznos *n*.
* Kad korisnik zada naredbu „ROLLBACK“ i želi odustati od promjena, vratite HP svim pokemonima na stanje kakvo je bilo u trenutku početka transakcije. Transakcija završava.
* Kad korisnik zada naredbu „COMMIT“ i želi potvrditi promjene, HP ostaje kakav jest i transakcija završava.
* Na kraju transakcije, ispišite sve pokemone.