OBJEKTNO ORIJENTISANO PROGRAMIRANJE

- domaći zadatak broj 2 -

Funkcionalna specifikacija

Na programskom jeziku C++ implementirati fajl sistem (*FileSystem*).

Učitavanje hijerarhije u fajl sistem

Implementirati operaciju `void FileSystem::loadHierarchy(const string& fs_filepath, const string& log_filepath);` koja u fajl sistem učitava informacije o hijerarhiji fajlova koja je opisana u datoteci čija je putanja data parametrom `fs_filepath`. Ukoliko je prethodno već učitana hijerarhija, sve informacije o staroj hijerarhiji fajl sistema se brišu i učitava se nova hijerarhija. Parametar `log_filepath` predstavlja putanju ka izlaznom log fajlu koji se koristi pri izvršavanju komande za ispis (videti u tabeli komandi). Svaki red ulazne datoteke sadrži punu putanju od korena fajl sistema do određenog foldera (npr. `\folder1\folder2`) ili fajla i sadržaja fajla koji je od imena fajla odvojen znakom razmaka (npr. `\folder1\folder2\file.txt hello world!\nHow are you?`). Novi redovi u sadržaju fajla se obeležavaju sa „`\n`". U hijerarhiji fajl sistema fajlom se smatra putanja koja se završava ekstenzijom (.txt, .exe …). Folderom se smatra putanja koja nema ekstenziju na svom kraju. Ulazna datoteka mora da sadrži putanje do svih elemenata fajl sistema! Element fajl sistema ne može da se nađe u delu putanje ka nekom drugom elementu ukoliko prethodno nije navedena putanja ka njemu.

primer 1 - ispravan ulaz	primer 2 - neispravan ulaz	primer 3 - neispravan ulaz

`\folder1`
`\folder2`
`\folder2\file1.txt Cao!\nCao!`
	`\folder1`
	`\folder2\file1.txt Cao!\nCao!`
		`\folder1`
		`\folder2\file1.txt Cao!\nCao!`
		`\folder2`

Elementi fajl sistema

Podržati sledeće elemente fajl sistema: foldere i fajlove.

Tipove fajlova koje treba podržati su tekstualni (.txt) fajl i izvršni fajl (.exe). Predvideti dodavanje novih vrsta fajlova. Predvideti da se u svakom folderu može naći proizvoljan broj foldera i fajlova. U nastavku su dati opisi svih vrsti fajlova:

- Fajl sa ekstenzijom ***.txt*** sadrži proizvoljan tekst.
- Fajl sa ekstenzijom ***.exe*** u svakom redu svog sadržaja sadrži poziv jedne komande. Komande koje se mogu naći u ***.exe*** fajlu su date u tabeli u nastavku. Predvideti dodavanje novih vrsta komandi.

naziv komande	oblik poziva komande	opis komande
LS	LS	Ispisuje u log fajl (čija je putanja zadata pri učitavanju hijerarhije fajl sistema) sadržaj tekućeg direktorijuma i svih njegovih poddirektorijuma u formatu ulazne datoteke. Pri ispisu svi elementi fajl sistema su alfabetski sortirani. Ne treba ispisivati sadržaj fajlova. Pr: \b.txt \folder2 \folder2\a.txt \folder2\folder3 \folder2\folder3\x.txt \folder2\folder3\z.txt \folder2\y.txt
CD	CD *naziv* CD ..	Postavlja tekući direktorijum u fajl sistemu na direktorijum dat nazivom *naziv* ili na roditeljski direktorijum tekućeg direktorijuma (drugi oblik poziva komande). Folder sa datim nazivom mora da se nalazi u tekućem direktorijumu. Smatrati da je tekući direktorijum pri kreiranju hijerarhije koreni direktorijum.

naziv komande	oblik poziva komande	opis komande
NEW	NEW *naziv_foldera* NEW *naziv_fajla.ekstenzija sadržaj*	Kreira nov folder/fajl u tekućem direktorijumu. Pri kreiranju fajla, drugi parametar komande je sadržaj fajla.
DEL	DEL *naziv_foldera* DEL *naziv_fajla.ekstenzija*	Briše folder/fajl sa datim nazivom u tekućem direktorijumu.
EXE	EXE *naziv*.exe	Izvršava komande u izvršnom fajlu sa nazivom *naziv*. Fajl mora da se nalazi u tekućem direktorijumu i mora biti tipa .exe.

Izvršavanje komandi nad fajl sistemom

Implementirati operaciju `void FileSystem::execute(const string& filepath);` koja izvršava sve komande date datotekom čija je putanja data parametrom `filepath`.

Obrada grešaka

Neregularne situacije obrađivati konceptom izuzetaka. Prilikom obrade greške, ispisati grešku na standardni izlaz. U slučaju greške prilikom izvršavanja komande, ispisati grešku u log fajl u formatu: `Error: oblik_poziva_komande_koja_je_izazvala_grešku\n`

Test funkcija

Javni test sadrži funkciju `void test();` koja testira rad sa fajl sistemom. Studentima je javno dostupna implementacija funkcije `test` i mogu da je menjaju da bi dodatno testirali svoj kod kao što je opisano komentarima u kodu. Studentima su dati ulazni fajlovi koji predstavljaju test primere kao i izlazni fajlovi koji predstavljaju očekivane izlaze izvršavanja komandi nad fajl sistemom radi mogućnosti provere.

Tehnički zahtevi i smernice za izradu rešenja

Programski sistem realizovati tako da bude detaljno komentarisan, modularan i lako proširiv novim klasama i operacijama. Klasa *FileSystem* i njene operacije moraju biti imenovane prema zahtevima domaćeg zadatka. Programski kod klasa rasporediti u odgovarajuće **.h** i **.cpp** fajlove. Iz kolekcije standardne biblioteke dozvoljeno je koristiti sledeće tipove: `<string>`, `<vector>`, `<list>`, `<stack>`, `<queue>`. Ukoliko u zadatku nešto nije dovoljno jasno definisano, treba usvojiti razumnu pretpostavku i na temeljima te pretpostavke nastaviti izgrađivanje svog rešenja.

21.12.2020. godine sa predmeta