Sveučilište u Zagrebu
Fakultet organizacije i informatike
Pavlinska 2, 42000 Varaždin

Obrazac za zadaću na predmetu "**Uzorci dizajna**" ak. god. 2025./2026.

Ime i prezime studenta/ice: Ivan Beusan

Matični broj: 0016150244

Dio A. Osnovni podaci o zadaći

R.br.	Pitanje	Odgovor	
1.	Grupa na seminaru:	G1	
2.	Broj i naziv zadaće:	DZ-3	Turistička agencija
3.	Procjena vremena za realizaciju bez decimala):	30 sati	
4.	Procjena % završenosti (bez decimala):	95 / 100%	
5.	Procjena bodova za izradu zadaće (1 decimala):	14 / (DZ3 - 15)	
6.	Žalim prezentirati zadaću:	NEMA PREZENTACIJE ZADAĆE!	
7.	Koji dijelovi iz opisa zadaće nisu realizirani:	-	
8.	Postoji li dio zadaće koji vrijedi posebno istaknuti i zašto:	-	
9.	Postoje li dijelovi zadaće koji imaju pogrešku u radu i koje:	-	
10.	Da li ste koristili tuđi programski kod u realizaciji zadaće izvan spomenutih izvora na nastavi:	Ako se to klasificira kao tuđi kod koristio sam generički tip u jednom Factory Methodu UcitavacPodataka što sam vidio na stack overflowu i činilo mi se korisno te je komentirano u kodu i u fusnoti ispod [1]	
11.	Da li ste koristili programska rješenja ili dijelove programskog koda od drugih kolega:	Ne.	
12.	Da li ste koristili alat/e generativne umjetne inteligencije u izradi zadaće, ako jeste koji/e ste koristili i za koje potrebe ste ga/ih koristili:	Da. Za eventualno istraživanje i raspravu o idejama za implementaciju uzoraka s obzirom da je puno brže i efikasnije od googleanja unedogled. AI se pokazao relativno nepouzdan za prepoznavanje potrebe za nekim uzorkom ali dobar za raspravu. Također je korišten za generiranje formata za tablične ispise te za eventualni troubleshooting, dijagnostiku greški i ispisa greški u terminalu.	

[1] https://docs.oracle.com/javase/tutorial/java/generics/why.html
https://stackoverflow.com/questions/39386586/c-sharp-generic-interface-and-factory-pattern

Dio B.1. Dokumentacija rješenja 2. zadaće (kopirano i nepromijenjeno)

Naziv uzorka dizajna	Klase koje sudjeluju u uzorku dizajna i u kojim ulogama	Status[1]	Opis razloga odabira uzorka dizajna
Builder	AranzmanBuilder (Builder), Aranzman (Product), AranzmanDirector (Director), Aplikacija i KomandaUp (Client)	P	P – dodan director po uputi nastavnika te je provedeno refaktoriranje zbog podjele cijelog sustava na dva modula. Builder je korišten za postupno i čitljivo sastavljanje složenog objekta Aranzman s puno atributa. Director centralizira "recept" izgradnje iz ulaznih podataka, a klijenti (Aplikacija/KomandaUp) više ne moraju ručno pozivati sve postaviX metode.
Builder	RezervacijaDirector (Director), Rezervacija (Product), Aplikacija i KomandaUp (Client)	P	P – dodan director po uputi nastavnika te je provedeno refaktoriranje zbog podjele cijelog sustava na dva modula. Director za Rezervacija standardizira izgradnju rezervacije iz podataka dobivenih iz liba i uklanja dupliciranje logike mapiranja po više mjesta (npr. Aplikacija i UP R). Time je konstrukcija konzistentna i jednostavnija za održavanje.
Factory Method	UcitavacFactory (Creator/Factory), UcitavacPodataka<T> (Product sučelje), CitacAranzmana i CitacRezervacija (ConcreteProduct), DatotekeFacadeImpl (Client)	S	Factory stvara odgovarajući učitavač (čitač) ovisno o vrsti podataka (aranžmani/rezervacije). Time se uklanja nepotrebno grananje i olakšava dodavanje potencijalnih novih čitača bez promjene klijentske logike.
Factory Method	KomandaFactory (Creator/Factory), Komanda (Product sučelje), KomandaXxxx (ConcreteProduct: KomandaItak/Itap/ Itas/Irta/Iro/Ip/Bp/ Ota/Orta/Drta/Up/ Audit/Q/Nepoznata...), Komande (Client)	P	Factory dinamički mapira unos korisnika na točnu komandu. Time je sustav proširiv te se izbjegava ogroman if/else lanac po aplikaciji. (Napomena iz komentara prve zadaće: factory ne smije vraćati null te vraća KomandaNepoznata (nije null object).
Decorator	AuditKomandaDecorator (ConcreteDecorator), KomandaDecorator (bazna Decorator klasa), Komanda (Component), KomandaXxx (ConcreteComponent), AuditDnevnik/AuditSpremnik (spremište audita), KomandaFactory (mjesto „wrappanja")	N	Dodana je funkcionalnost audita bez mijenjanja postojećih komandi gdje se komanda „wrappa" dekoratorom koji za izvršenja komande bilježi događaj. Time se poštuje open-closed princip te originalne komande ostaje netaknute, a audit se može koristiti centralno u factoryju.
Bridge	FormatIspisaBridge (Implementor sučelje), TablicniFormat (ConcreteImplementor), KomandaXxx (Client)	P	Bridge je odabran jer odvaja apstrakciju ispisa (FormatIspisaBridge) od konkretne implementacije implementacije (Što vs. Kako se nešto ispisuje.) Tablični format je time otvoren za promjene tj. mogu se dodati novi formati ispisa bez mijenjanja postojeće implementacije. Metoda ispisa ne bi smjela imati Adapter kao argument u smislu krive uloge te je to izmijenjeno u usporedbi sa DZ1.
State	StanjeAranzmana (State sučelje), StanjeUPripremiAranzman/	N	State uzorak obuhvaća ponašanje stanja aranžmana. Aranzman je Context i

[1] N – dodan u 2. zadaći, P – promijenjen u 2. zadaći, S – bez promjena u 2. zadaći

	StanjeAktivanAranzman/ StanjePopunjenAranzman/ StanjeOtkazanAranzman (ConcreteState), Aranzman (Context)		prebacuje stanje na temelju broja prijava/aktivnih mjesta odnosno kad se pokrene rekalkulacija stanja uz pravilo da "otkazan" ostaje nepromjenjivo stanje.
State	StanjeRezervacije (State sučelje), StanjePrimljenaRezervacija/ StanjeAktivnaRezervacija/ StanjeNaCekanjuRezervacija/ StanjeOdgodenaRezervacija/ StanjeOtkazanaRezervacija/ StanjeNovaRezervacija (ConcreteState), Rezervacija (Context)	N	State uzorak obuhvaća stanja rezervacije (primljena/aktivna/na čekanju/odgođena/otkazana...). Time se pravila promjene stanja mogu održavati bez milijun provjera s if i else.
Facade	DatotekeFacade (Facade sučelje), DatotekeFacadeImpl (ConcreteFacade), CitacAranzmana/ CitacRezervacija /UcitavacFactory/ CsvParser (Subsystem), Aplikacija i KomandaUp (Client)	N	Klijent (app modul) ne smije direktno ovisiti o lib internim klasama. Facade daje stabilan API (ucitajAranzmane/ucitajRezervacije) i skriva detalje parsiranja, validacije i tvorničke logike od appa koji dobije validirane podatke od liba.
Composite	ElementRezervacijskeStrukture (Component sučelje), Aranzman (Composite), Rezervacija (Leaf)	N	Rezervacije su hijerarhijski pod aranžmanoma, a Aranzman sadrži listu Rezervacija i predstavlja kompozitnu strukturu. To omogućuje operacije nad cijelim aranžmanom poput brisanja i statistike te iteriranje nad djecom bez da klijent zna detalje spremanja.
Adapter	IspisAranzmanaAdapter (Adapter), IspisniRed (Target), Aranzman (Adaptee), TablicniFormat (koristi Target)	S	Adapter kojim se objekt Adapter prilagođava formatu tabličnog ispisa podataka. Logika ispisa se odvaja od domenskog modela te omogućuje isti način ispisa različitih modela.
Adapter	IspisAranzmanDetaljnoAdapter (Adapter), Aranzman (Adaptee), TablicniFormat/KomandaItap (Client)	S	Adapter kojim se omogućuje prikaz samo jednog atributa i njegovih podataka komandom ITAP.
Adapter	IspisRezervacijaAdapter (Adapter), IspisniRed (Target), Rezervacija (Adaptee), TablicniFormat (Client)	S	Adapter koji prilagođava objekt Rezervacija formatu tabličnog ispisa. Time se izbjegava da sama klasa Rezervacija ima metode za ispis.
Adapter	IspisRezervacijaOsobeAdapter (Adapter), IspisniRed (Target), Rezervacija+Aranzman (Adaptee), TablicniFormat (Client)	S	Adapter koji prilagođava ispis stvari iz objekata Aranžman i Rezervacije u tablični ispis te standardizira takav ispis.
Adapter	IspisAuditAdapter (Adapter), IspisniRed (Target), AuditStavka (Adaptee), TablicniFormat (Client)	N	Adapter pretvara audit zapis (AuditStavka) u formatiran ispisni red. Audit model ostaje čist, a formatiranje je izdvojeno iz poslovne logike i komande AUDIT.
Adapter	IspisAuditZbrojAdapter (Adapter), IspisniRed (Target), AuditDnevnik (Adaptee), TablicniFormat (Client)	N	Adapter služi za tablični prikaz spremljenih audit podataka. Time se podaci ne formatiraju za ispis u komandi nego se samo pretvore u ispisne redove.
Adapter	IspisStatistikaAranzmanaAdapter (Adapter), IspisniRed (Target), Rezervacije (Adaptee) , TablicniFormat (Client)	N	Adapter za ITAS prikaz statistike u tabličnom obliku

Dio B.2. Dokumentacija rješenja 3. zadaće

Naziv uzorka dizajna	Klase koje sudjeluju u uzorku dizajna i u kojim ulogama	Status[2]	Opis razloga odabira uzorka dizajna
Builder	AranzmanBuilder (Builder), Aranzman (Product), AranzmanDirector (Director), Aplikacija i KomandaUp (Client)	S	P – dodan director po uputi nastavnika te je provedeno refaktoriranje zbog podjele cijelog sustava na dva modula. Builder je korišten za postupno i čitljivo sastavljanje složenog objekta Aranzman s puno atributa. Director centralizira "recept" izgradnje iz ulaznih podataka, a klijenti (Aplikacija/KomandaUp) više ne moraju ručno pozivati sve postaviX metode.
Builder	RezervacijaDirector (Director), Rezervacija (Product), Aplikacija i KomandaUp (Client)	S	P – dodan director po uputi nastavnika te je provedeno refaktoriranje zbog podjele cijelog sustava na dva modula. Director za Rezervacija standardizira izgradnju rezervacije iz podataka dobivenih iz liba i uklanja dupliciranje logike mapiranja po više mjesta (npr. Aplikacija i UP R). Time je konstrukcija konzistentna i jednostavnija za održavanje.
Factory Method	UcitavacFactory (Creator/Factory), UcitavacPodataka<T> (Product sučelje), CitacAranzmana i CitacRezervacija (ConcreteProduct), DatotekeFacadeImpl (Client)	S	Factory stvara odgovarajući učitavač (čitač) ovisno o vrsti podataka (aranžmani/rezervacije). Time se uklanja nepotrebno grananje i olakšava dodavanje potencijalnih novih čitača bez promjene klijentske logike.
Factory Method	KomandaFactory (Creator/Factory), Komanda (Product sučelje), KomandaXxxx (ConcreteProduct: KomandaItak/Itap/ Itas/Irta/Iro/Ip/Bp/ Ota/Orta/Drta/Up/ Audit/Q/Nepoznata...), Komande (Client)	S	Factory dinamički mapira unos korisnika na točnu komandu. Time je sustav proširiv te se izbjegava ogroman if/else lanac po aplikaciji. (Napomena iz komentara prve zadaće: factory ne smije vraćati null te vraća KomandaNepoznata (nije null object).
Decorator	AuditKomandaDecorator (ConcreteDecorator), KomandaDecorator (bazna Decorator klasa), Komanda (Component), KomandaXxx (ConcreteComponent), AuditDnevnik/AuditSpremnik (spremište audita), KomandaFactory (mjesto „wrappanja")	S	Dodana je funkcionalnost audita bez mijenjanja postojećih komandi gdje se komanda „wrappa" dekoratorom koji za izvršenja komande bilježi događaj. Time se poštuje open-closed princip te originalne komande ostaje netaknute, a audit se može koristiti centralno u factoryju.
Bridge	FormatIspisaBridge (Implementor sučelje), TablicniFormat (ConcreteImplementor), KomandaXxx (Client)	S	Bridge je odabran jer odvaja apstrakciju ispisa (FormatIspisaBridge) od konkretne implementacije implementacije (Što vs. Kako se nešto ispisuje.) Tablični format je time otvoren za promjene tj. mogu se dodati novi formati ispisa bez mijenjanja postojeće implementacije. Metoda ispisa ne bi smjela imati Adapter kao argument u smislu krive uloge te je to izmijenjeno u usporedbi sa DZ1.
State	StanjeAranzmana (State sučelje), StanjeUPripremiAranzman/ StanjeAktivanAranzman/	S	State uzorak obuhvaća ponašanje stanja aranžmana. Aranzman je Context i prebacuje stanje na temelju broja prijava/aktivnih mjesta

[2] N – dodan u 3. zadaći, P – promijenjen u 3. zadaći, S – bez promjena u 3. zadaći

	StanjePopunjenAranzman/ StanjeOtkazanAranzman (ConcreteState), Aranzman (Context)		odnosno kad se pokrene rekalkulacija stanja uz pravilo da "otkazan" ostaje nepromjenjivo stanje.
State	StanjeRezervacije (State sučelje), StanjePrimljenaRezervacija/ StanjeAktivnaRezervacija/ StanjeNaCekanjuRezervacija/ StanjeOdgodenaRezervacija/ StanjeOtkazanaRezervacija/ StanjeNovaRezervacija (ConcreteState), Rezervacija (Context)	S	State uzorak obuhvaća stanja rezervacije (primljena/aktivna/na čekanju/odgođena/otkazana...). Time se pravila promjene stanja mogu održavati bez milijun provjera s if i else.
Facade	DatotekeFacade (Facade sučelje), DatotekeFacadeImpl (ConcreteFacade), CitacAranzmana/ CitacRezervacija /UcitavacFactory/ CsvParser (Subsystem), Aplikacija i KomandaUp (Client)	S	Klijent (app modul) ne smije direktno ovisiti o lib internim klasama. Facade daje stabilan API (ucitajAranzmane/ucitajRezervacije) i skriva detalje parsiranja, validacije i tvorničke logike od appa koji dobije validirane podatke od liba.
Composite	ElementRezervacijskeStrukture (Component sučelje), Aranzman (Composite), Rezervacija (Leaf)	P	Rezervacije su hijerarhijski pod aranžmanoma, a Aranzman sadrži listu Rezervacija i predstavlja kompozitnu strukturu. To omogućuje operacije nad cijelim aranžmanom poput brisanja i statistike te iteriranje nad djecom bez da klijent zna detalje spremanja.
Command	Komanda(Interface), KomandaIrta….itd. (Konkretna komanda), Komande (Invoker), UpraviteljAranžmanima…itd. (Receiver), KomandaFactory (Client)	N	Command je kostur interaktivnog sustava komandi gdje konzola samo pokreće objekte-komande, a one rade stvarni posao u programu.
Strategy	StrategijaOgranicenjaRezervacija(sučelje), StartegijaJdr(ConcreteStrategy), UpraviteljRezervacijama(Context), Aplikacija(Client)	N	Pravilo --jdr je zaseban algoritam koji se može mijenjati bez diranja UpraviteljRezervacijama. Omogućuje da se u runtime-u argumentima bira ponašanje (--jdr), bez if/else grananja kroz cijeli sustav.
Strategy	StrategijaOgranicenjaRezervacija(sučelje), StrategijaVdr(ConcreteStrategy), UpraviteljRezervacijama(Context), Aplikacija(Client)	N	--vdr je drugi algoritam s drugačijim pravilima. Strategy omogućuje da se uključi alternativni algoritam bez promjene ostatka koda.
Strategy	StrategijaOgranicenjaRezervacija(sučelje), StrategijaBezOgranicenja(ConcreteStrategy), UpraviteljRezervacijama(Context), Aplikacija(Client)	N	Null-object" strategija s kojom sustav radi normalno bez posebnih ograničenja, bez provjera je li strategija postavljena a što se tiče ograničenja rezervacija gledaju se samo min i max broj rezervacija za aranžman.
Visitor	Posjetitelj(Visitor), Posjetljiv(Element), Aranzman(ConcreteElement), Rezervacija(ConcreteElement)	N	Omogućuje dodavanje novih operacija nad strukturom aranžman -> rezervacije bez mijenjanja istih.
Observer	Aranzman(Subject), Pretplatnik(Observer), OsobaPretplatnik(ConcreteObserver), UpraviteljRezervacijama(Client/okidač)	N	Pretplatnici su automatski obaviješteni o promjenama bez da UpraviteljRezervacijama zna detalje tko su pretplatnici i kako reagiraju. Observer tako uklanja čvrstu vezu između logike promjene I konkretnog primatelja.
Memento	AranzmanMemento(Memento), SpremisteAranzmana(Caretaker), Aranzman(Originator)	N	Memento omogućuje vraćanje na prethodno stanje aranžmana te zadržava enkapsulaciju tako da ostatak sustava ne mora znati kako se točno unutarnje stanje aranžmana dobiva. Dobije snapshot i može ga vratiti.

Chain of Responsibility	Komande(Client), HandlerUnosa(Handler), AptraktniHandlerUnosa(Abstract handler), (Concrete handleri): TrimWhitespaceHandler, PrazanHandler, TokenizacijaHandler, KanonizirajKomanduHandler, KanonizirajArgumenteHandler, ValidacijaQHandler, QIzlazHandler	N	Umjesto da postoji jedan "mega parser" s puno if/else slučajeva, koristi se niz malih koraka koji su lako proširivi i jasno odvojeni po odgovornosti.
Adapter	IspisAranzmanaAdapter (Adapter), IspisniRed (Target), Aranzman (Adaptee), TablicniFormat (koristi Target)	S	Adapter kojim se objekt Adapter prilagođava formatu tabličnog ispisa podataka. Logika ispisa se odvaja od domenskog modela te omogućuje isti način ispisa različitih modela.
Adapter	IspisAranzmanDetaljnoAdapter (Adapter), Aranzman (Adaptee), TablicniFormat/KomandaItap (Client)	S	Adapter kojim se omogućuje prikaz samo jednog atributa i njegovih podataka komandom ITAP.
Adapter	IspisRezervacijaAdapter (Adapter), IspisniRed (Target), Rezervacija (Adaptee), TablicniFormat (Client)	S	Adapter koji prilagođava objekt Rezervacija formatu tabličnog ispisa. Time se izbjegava da sama klasa Rezervacija ima metode za ispis.
Adapter	IspisRezervacijaOsobeAdapter (Adapter), IspisniRed (Target), Rezervacija+Aranzman (Adaptee), TablicniFormat (Client)	S	Adapter koji prilagođava ispis stvari iz objekata Aranžman i Rezervacije u tablični ispis te standardizira takav ispis.
Adapter	IspisAuditAdapter (Adapter), IspisniRed (Target), AuditStavka (Adaptee), TablicniFormat (Client)	N	Adapter pretvara audit zapis (AuditStavka) u formatiran ispisni red. Audit model ostaje čist, a formatiranje je izdvojeno iz poslovne logike i komande AUDIT.
Adapter	IspisAuditZbrojAdapter (Adapter), IspisniRed (Target), AuditDnevnik (Adaptee), TablicniFormat (Client)	N	Adapter služi za tablični prikaz spremljenih audit podataka. Time se podaci ne formatiraju za ispis u komandi nego se samo pretvore u ispisne redove.
Adapter	IspisStatistikaAranzmanaAdapter (Adapter), IspisniRed (Target), Rezervacije (Adaptee) , TablicniFormat (Client)	N	Adapter za ITAS prikaz statistike u tabličnom obliku

Dio C.1. Opis promjena u odnosu na prethodnu zadaću

- Dodana nova vlastita funkcionalnost **bez** uvođenja nove komande - Obrada korisničkog unosa kroz lanac prije nego što se kreira i izvrši komanda. Funkcionalnost nije nova komanda, nego poboljšanje postojećeg toka interakcije: unos se normalizira, validira i priprema na jednom mjestu. Koristi se Chain of Responsibility uzorak dizajna.
- Unos se trimma i normaliziraju se razmaci, prazni unosi se ignoriraju s Null objectom. Npr. može se upisati „IRTA 3 ČOPADO" i to će program ispraviti ako su zaista viška samo razmaci i tabovi.
- Iz programa se može izaći sa q, quit i exit.
- Strategy je iskorišten za odabir načina kalkulacije statusa dodanih rezervacija.
- Command se koristi za odvajanje unosa i parsiranja od logike te invoker uvijek na isti način može izvršiti bilo koju komandu.
- Visitor se koristi kako bi se dodale nove funkcinlanosti bez mijenjanja ostalog koda u oovm slučaju dodaje se pretraga aranžmana i rezervacija po unesenom stringu.
- Memento se koristi za spremanje stanja nekog aranžmana i kasnije vraćanje istog.
- Observer se korsiti za mogućnost pretplate nekog korisnika na promjene u nekom aranžmanu bilo na promjene vezane za aranžman ili rezervacije koje su u njemu.
- Popravljen Composite jer se u DZ-2 slučajno „išlo na sporedna vrata" no i dalje je zadržana generalna struktura Compositea. Sad je to riješeno.
- Eventualno još koji kozmetički popravci u ispisima.

Dio C.2. Opis funkcionalnosti za uzorak dizajna Decorator (kopirano i nepromijenjeno iz 2. zadaće)

Decorator je iskorišten da se postojećim komandama doda audit bez izmjene njihovih klasa. Klase koje sudjeluju u uzorku: AuditKomandaDecorator (ConcreteDecorator), KomandaDecorator (bazna Decorator klasa), Komanda (Component), KomandaXxx (ConcreteComponent), AuditDnevnik (spremište audita), KomandaFactory (mjesto „omotavanja").
Komanda je Component (sučelje koje sve komande implementiraju), a sve konkretne komande (KomandaItak, KomandaUp, KomandaOta, …) su ConcreteComponent. KomandaDecorator je Decorator te implementira Komanda i u sebi drži referencu na "omotanu" komandu. AuditKomandaDecorator je ConcreteDecorator koji prije izvršavanja komande (izvrsi()) pokrene mjerenje vremena, a iza izvršavanja ga zaustavlja i zapisuje u AuditDnevnik koji služi kao mjesto spremanja svih zapisa. Također u zapis upisuje naziv komande i njene parametre, vrijeme izvršavanja, trajanje izvršavanja te status izvršavanja (OK). Decorator se primjenjuje u tvornici komandi , a audit funkcionalnost se dodaje na već postojeće klase komandi bez njihovog mijenjanja omotavanjem oko njih.

Dio C.3. Opis funkcionalnosti za uzorak dizajna Chain-of-Responsibility

Implementirana je funkcionalnost obrada korisničkog unosa putem lanca odgovornosti prije kreiranja i izvršavanja komande. Cilj je da se unos očisti i normalizira (razmaci, trim), prepozna kao prazan ili da se želi izaći iz programa, razbije na naredbu i argumente, kanonizira u standardni oblik (uppercase), validira (npr. sprječavanje izvršavanja "Q nešto") te vrati kao strukturirani unos.
Obrada unosa prirodno ide u sekvencijalnim koracima te svaki korak treba imati mogućnost da obavi jednu malu odgovornost i po potrebi prekine daljnju obradu (npr. prazno, izlaz, greška). Također omogućuje se lako dodavanje novih koraka bez mijenjanja postojećih. Klijent pokreće obradu te apstraktni handler drži referencu na slijedeći element u lancu te prekida obradu ako dobije greska/ignoriraj/izlaz.

Konkretni handleri:
TrimIWhitespaceHandler: normalizira liniju unosa (trim i razmaci).
PrazanUnosHandler: ako je linija prazna ignorira ju i zaustavlja obradu.
QIzlazHandler: ako je linija q/quit/exit poziva izlaz iz programa.
TokenizacijaHandler: parsira liniju unosa u naredbu i argumente.
KanonizirajKomanduHandler: pretvara naredbu u UPPERCASE.
ValidacijaQHandler: ako je unesen Q uz argumente daje grešku.
KanonizirajArgumenteHandler: ciljano kanonizira argumente za komande (BP/UP/PPTAR, IRTA, ITAP/ITAK).

UnosKontekst je objekt koji „putuje" kroz lanac te sadrži sve podatke i stanje obrade (linija, naredba, argumenti, greska, izlaz, ignoriraj).
Što se tiče dobrobiti za ovaj projekt CoR pridonosi tako da komande postaju otpornije na različit način unosa (razmaci, tabovi itd.), izlaz iz programa je fleksibilniji (q, quit, exit), greške vezane uz format Q su jednoznačne i centralizirane, sve se događa prije kreiranja komande.

Dio D. Dijagram klasa s naglašavanjem klasa koje sudjeluju u pojedinom uzorku dizajna