Zadaci za vežbe iz predmeta
Softverski obrasci i komponente
Python zadaci za vežbanje

1 Zadaci funkcije

Kreirati novi projekat koji će imati naziv Vezbe1Zadaci. Unutar tog projekta dodati paket zadaci1. U paketu zadaci1 dodati modul funkcije.py.

1.1 Zadatak 1

U modulu funkcije.py definisati novu funkciju brojanje_reci koja očekuje jedan parametar listu reči. Unutar funkcije, na osnovu dobijene liste reči, kreirati listu čiji n-ti član odgovara dužini n-te reči prve liste. Kao povratnu vrednost funkcije vratiti dobijenu listu sa dužinama. Napraviti listu proizvoljnih reči koja će biti prosleđena funkciji, a rezultat prikazati na konzoli.

1.2 Zadatak 2

U modulu funkcije.py definisati novu funkciju switch koja treba da simulira datu switch naredbu pomoću rečnika i njegove get metode sa default-nom vrednošću.

```
String string;
switch(x){
        case 10:
                string = "deset";
                break;
        case 9:
                string = "devet";
                break;
        case 8:
                string = "osam";
                break;
        case 7:
                string = "sedam";
                break;
        case 6:
                string = "sest";
                break;
        default:
                string = "nije polozeno";
                break;
}
System.out.println(string);
```

1.3 Zadatak 3

U paketu zadaci1 dodati novi modul parametri.py. U modulu parametri.py dodati tri posebne funkcije:

a) prva funkcija očekuje parametre duzina, sirina i visina za koje su definisane default-ne vrednosti 1

b) druga funkcija koja očekuje promenljivu vrednost parametara koristeći *args param-etar

c) treća funkcija koja očekuje promenljivu vrednost parametara koristeći **kwargs parametar

Sve tri funkcije treba da preuzmu vrednosti parametara duzina, sirina i visina, provere da su prosleđene vrednosti veće od 0 i vrate te tri vrednosti. Prilikom poziva funkcija proslediti samo duzinu, samo sirinu ili sve tri vrednosti za duzinu, sirinu i visinu.

2 Zadaci import

2.1 Zadatak 1

U paketu zadaci1 dodati novi modul import1.py. Unutar tog modula pozvati funkcije iz modula funkcije.py i parametri.py koristeći samo import naredbu.

2.2 Zadatak 2

U paketu zadaci1 dodati novi modul import2.py. Unutar tog modula pozvati funkcije iz modula funkcije.py i parametri.py koristeći from naredbu za import-vanje modula.

2.3 Zadatak 3

U paketu zadaci1 dodati novi modul import3.py. Unutar tog modula pozvati funkcije iz modula funkcije.py i parametri.py koristeći from naredbu za import-vanje modula, ali korišćenjem relativne putanje do modula funkcije.py i parametri.py.
Da biste pokrenuli ovaj primer potrebno je u paket zadaci1 dodati novi modul pokretanje_import3.py i sa import naredbom import-ovati modul import3.py.

Napomena: Relativna putanja se navodi sa .:
from . import modul
tačka određuje da se koristi putanja paketa u kojem se trenutni modul nalazi. Modul u kojem se koristi relativni import ne treba koristiti kao glavni modul iz kojeg se pokreće aplikacija, jer se za relativni import koristi putanja koja je postavljena u __name__ atributu modula.

2.4 Zadatak 4

Direktno u projektu Vezbe1Zadaci dodati novi modul import4.py. Unutar tog modula pozvati funkcije iz modula funkcije.py i parametri.py tako što ćete import-ovati sve module iz paketa zadaci1.
Napomena: Da bi se omogućilo importovanje svih modula iz nekog paketa, nije dovoljno samo pozvati from paket import *. Neophodno je prvo definisati u __init__.py unutar paketa spisak svih njegovih sadržanih modula koji se importuju na sledeći način:
__all__ = ["modul1","modul2","modul3"]

3 Zadaci klase

U projekat Vezbe1Zadaci dodati paket zadaci2. U ovom paketu dodati modul model.py.

3.1 Dodavanje klase Identifikacija

U modulu model.py definisati klasu Identifikacija sa atributima:

a) oznaka koji čuva tekstualnu vrednosti

b) opis koji čuva teksutalnu vrednost

Redefinisati __str__ metodu tako da prikazuje vrednosti svih atributa. Koristiti format funkciju za prikaz vrednosti.

3.2 Dodavanje klase Dimenzije

U modulu model.py definisati klasu Dimenzije sa atributima:

a) duzina koji čuva decimalne vrednosti

b) sirina koji čuva decimalne vrednosti

c) visina koji čuva decimalne vrednosti

Za svaki atribut klase Dimenzije, prilikom dodele vrednosti proveriti da se dodeljuje vrednost veća od 0, a u suprotnom izazvati izuzetak. Proveru implementirati u posebni get i set metodama. Za definisanje get i set metoda koristiti dekoratore property i setter.
Redefinisati __str__ metodu tako da prikazuje vrednosti svih atributa. Koristiti format funkciju za prikaz vrednosti.

3.3 Dodavanje klase Deo

U modulu model.py definisati klasu Deo i za nju definisati da nasleđuje klase Identifikacija i Dimenzije.
Redefinisati __str__ metodu tako da prikazuje vrednosti svih atributa iz klasa Identifikacija i Dimenzije. Koristiti format funkciju za prikaz vrednosti.
Napomena: Potrebno je klasu Identifikacija proširiti da se u konstruktoru za prosleđivanje dodatnih parametara sirina, duzina i visina koristi **kwargs parametar.

3.4 Dodavanje klase MehanickiDeo

U modulu model.py definisati klasu MehanickiDeo sa atributom:

a) tezina koji čuva decimalne vrednosti

Za klasu MehanickiDeo definisati da nasleđuje klasu Deo.
Redefinisati __str__ metodu tako da prikazuje vrednosti svih atributa. Koristiti format funkciju za prikaz vrednosti.

3.5 Dodavanje klase ElektricniDeo

U modulu model.py definisati klasu ElektricniDeo sa metodom:

a) elektricna_potrosnja preko koje se računa električna potrošnja dela i koja vraća decimalnu vrednost

Za metodu elektricna_potrosnja odrediti da je abstraktna metoda koristeći dekorator abstractmethod. Za klasu ElektricniDeo definisati da nasleđuje klasu Deo i klasu ABC.
Napomena: Klasu ABC i dekorator abstractmethod treba importovati iz python modula abc.

3.6 Dodavanje klase Okvir

U modulu model.py definisati klasu Okvir i u njoj definisati atribut:

a) tip_materijala koji čuva tekstualne vrednosti

Za klasu Okvir definisati da nasleđuje klasu MehanickiDeo.
Redefinisati __str__ metodu tako da prikazuje vrednosti svih atributa. Koristiti format funkciju za prikaz vrednosti.

3.7 Dodavanje klase Motor

U modulu model.py definisati klasu Motor i u njoj definisati atribute:

a) vreme_rada koji čuva celobrojnu vrednost

b) obrtaja_u_minuti koji čuva decimalnu vrednost

c) potrosnja_po_obrtaju koji čuva decimalnu vrednost

Za klasu Motor definisati da nasleđuje klase MehanickiDeo i ElektricniDeo. Redefinisati metodu elektricna_potrosnja tako da racuna električnu potrošnju kao proizvod vremena rada, obrtaja u minuti i potrošnje po obrtaju.
Redefinisati __str__ metodu tako da prikazuje vrednosti svih atributa. Koristiti format funkciju za prikaz vrednosti.

3.8 Dodavanje klase Senzor

U modulu model.py definisati klasu Senzor i u njoj definisati atribute:

a) tip koji čuva tekstualnu vrednost (toplotni, svetlosni ...)

b) merna_jedinica koji čuva tekstualnu vrednost

c) izmerena_vrednost koji čuva decimalnu vrednost

d) potrosnja_po_merenju koji čuva decimalnu vrednost

e) broj_merenja koji čuva decimalnu vrednost

Za klasu Senzor definisati da nasleđuje klasu ElektricniDeo. Redefinisati metodu elektricna_potrosnja tako da racuna električnu potrošnju kao proizvod potrošnje po merenju i broja merenja.
Redefinisati __str__ metodu tako da prikazuje vrednosti svih atributa. Koristiti format funkciju za prikaz vrednosti.

3.9 Instanciranje objekata

U paketu zadaci2 dodati modul test.py. U ovom modulu instancirati objekte klasa:

a) Okvir

b) Motor

c) Senzor