|  |
| --- |
| **NAZIV OBJEKTA ZNANJA \*** |
|  |

|  |  |
| --- | --- |
| **PROLOG** | |
| **Autor \*** | *Nemanja Stojanović* |
| [**Klasifikacija**](http://dita.metropolitan.ac.rs:8880/Preview/)**\*** | IT2008-IPT-SCRP-STRL-V-V05 |
| **Težina \*** | Srednja |
| **Ključne reči \*** | skripting jezici, perl,objekti, paketi |
| **Ko sluša \*** | OAS student |
| **Trajanje \*** | 2h |
| **Komentari autora** |  |

|  |
| --- |
| **SADRŽAJ** |
| **Apstrakt** |
|  |
| **Cilj \*** |
| Upoznavanjen sa Perl objektnim sistemom, definisanjem klasa i modula. |
| **Uvodne napomene** |
|  |
| **Naziv sekcije(1)** |
| ***Perl - packages and objects*** |
| **Sadržaj sekcije(1)** |
| **Kreiranje klasa**  U perlu su klase paketi, dok su metode podrutine. Perl nema klase kao sintaksni konstrukt, ali omogućava njihovo definisanje i instanciranje pomoću paketa. Počećemo sa definicijom jednostavne klase Animal koja ima polje "text" i metodu "speak", koja ispisuje taj tekst.  use v5.10.0;  use warnings;  use strict;  package Animal;  sub new {  my $class = $\_[0];  my $self = {text => "I am an abstract animal.\n"};  return bless($self, $class);  }   * Definisali smo konstruktor (new) * Prvi parametar pri instanciranju je ime klase * Self je hash objekat koji sami kreiramo * Funkcija bless vezuje referencu za određenu klasu, čineći je referencom na instancu te klase   Ovo se može napisati jednostavnije, obzirom da je @\_ podrazumevani argument za većinu funkcija:  sub new {  return bless({text => "I am an abstract animal.\n"}, shift);  }  **Dodavanje metoda**  Na klasu Animal dodaćemo metodu "speak", koja štampa polje "text".  sub speak {  my $self = shift;  print $self->{"text"}  }  Objašnjenje:   * za instance metode (metode koje se pozivaju nad instancom objekta), prvi argument je referenca na sam objekat (u Javi bi to bila this referenca), koji se uzima iz liste argumenata funkcijom shift. * $self je referenca (u ovom slučaju referenca na hash koji je vezan za Animal klasu) * referenca $self se mora de-referencirati da bi se pristupilo njenim vrednostima, što se može učiniti operatorom '->' . * $self->{"text"} dereferencira $self, pristupa hash-u sa ključem "text" * vrednost poslednje izvršenog izraza je povratna vrednost metode, u ovom slučaju vrednost polja text.   **Nasleđivanje**  Za nasleđivanje se u Perl-u koristi klasna promenljiva @ISA, koja je array parent klasa.  package Duck;  our @ISA = qw(Animal); # qw(Animal) je ekvavilentno ("Animal")  sub new {  return bless({text => "I am a a duck.\n"}, shift);  }  Klasa Duck je sada pod-klasa klase Animal. U podklasi je moguće redefinisati metode iz roditeljske klase.  **Finalizacija modula**  Da bi Perl učitao modul, potrebno je da poslednja vrednost u modulu bude true. Ovo se obično radi tako što se na poslednju liniju postavi "1;". Perl moduli se čuvaju kao tekstualni fajlovi sa ekstenzijom .pm. |
| **Naziv sekcije(2)** |
| ***Instanciranje klasa i pozivanje metoda*** |
| **Sadržaj sekcije(2)** |
| Komandom "use ImeModula;", učitavate perl modul u trenutni kontekst. Ovo je slično import i require pozivima u Python-u i Ruby-u.  # Instanciranje objekata:  use Objects;  my $animal = Animal->new;  $animal->speak; # Ispisuje "I am an abstract animal.\n"  $animal = Duck.new;  $animal->speak; # Ispisuje "I am a a duck.\n"  Instanca klase se vraća iz "new" funkcije kao referenca, pa je potrebno dereferencirati je radi poziva metoda ili pristupa poljima. Sama referenca je sklar, pa se čuva u promenljivama prefiksovanim znakom $. |

|  |  |
| --- | --- |
| **LINKOVI SA SLIČNIM SADRŽAJEM** | |
| **Opis** | **URL** |
|  |  |

|  |  |
| --- | --- |
| **REČNIK POJMOVA I SKRAĆENICA** | |
| **Termin** | **Opis** |
|  |  |
|  |  |

|  |
| --- |
| **PODOBJEKTI ZNANJA (slučajevi korišćenja, procedure korak po korak, reference, itd.)** |
| **Naslov podobjekta znanja** |
|  |
| **Sadržaj podobjekta znanja** |
|  |

|  |
| --- |
| **OSTALI PREDLOŽENI SADRŽAJ OD STRANE AUTORA** |
| **Naslov** |
|  |
| **Sadržaj** |
|  |