

TIPOVI PODATAKA

Pojam tipa

Statička i dinamička tipizacija

Slaba i jaka tipizacija

Ekvivalentnost tipova

Tip podataka

- Tip podataka određuju sledeći elementi:
 - Skup vrednosti koje mogu biti predstavljene
 - Format registrovanja podataka
 - Skup operacija koje se nad podacima tog tipa mogu izvršavati

Provera tipova (*type checking*)

- Provera tipova (*type checking*) podrazumeva da se pre izvršenja bilo koje operacije vrši provera da li su operandi koji u njoj učestvuju odgovarajućeg tipa
 - Primer:
 - Aritmetičke operacije (+, -, *, /) se mogu izvoditi samo nad operandima numeričkog tipa
 - Logičke operacije (and, or, xor,...) se mogu izvoditi samo nad podacima logičkog tipa
 - Indeks polja mora biti ceo broj
 -

Provera tipova (*type checking*)

- Zavisno od toga, kada su informacije o tipu promenljivih (pa samim tim i o tipu izraza) poznate, programske jezike delimo na:
 - Programske jezike sa **statičkom tipizacijom**
 - Programske jezike sa **dinamičkom tipizacijom**

Statička tipizacija

- Tipovi promenljivih su konstantni za sve vreme izvršenja programa i poznati su u toku prevođenja
 - Svaka promenljiva mora biti deklarirana pre njenog korišćenja
- Provera tipova se vrši u vreme prevođenja programa (*compile-time type checking*)
- Jezici sa statičkom tipizacijom
 - C, C++, Java, C#

Dinamička tipizacija

- Tipovi promenljivih poznati tek u vreme izvršenja programa i mogu se menjati u toku izvršenja programa
 - Promenljive se ne deklarišu u programu
- Provera tipova se vrši u vreme izvršenja programa (*run-time type checking*)
- Jezici sa dinamičkom tipizacijom
 - Perl, PHP, Python, JavaScript

Statička i dinamička tipizacija - primer

- C:

```
int a, b;  
a = 10;  
b = 5;  
b = a + b;
```

- Python:

```
a = 10;  
b = 5;  
b = a + b;  
a = "Danas ";
```

Statička i dinamička tipizacija - poredjenje

- Statička:

- Provera tipova efikasnija (radi se samo jednom u toku kompajliranja)
- Provera tipova sigurnija

- Dinamička:

- Provera tipova u toku izvršenja programa što usporava izvršenje
- Zahteva „etiketiranje“ promenljivih u toku izvršenja (pamćenje njihovih tipova) što povećava zahtevani memorijski prostor
- Fleksibilnost u korišćenju promenljivih (što je poželjno kad tipovi nekih podataka nisu unapred poznati)

Prednost se daje statičkoj tipizaciji kad god je to moguće.

Prelaz sa dinamičke na statičku tipizaciju u skript jezicima

JavaScript → TypeScript

Problem mešovitih izraza

- Mešoviti izrazi – izrazi u kojima se pojavljuju operandi različitih tipova.
- Primer:

```
double x;  
int i, j, k;  
i = x;  
k = x * j;
```

- Kako će ovaj kod biti protumačen zavisi od koncepta tipova podataka koji je u jeziku korišćen
- 2 osnovna koncepta tipizacije:
 - Slaba tipizacija (*weakly typed languages*)
 - Jaka tipizacija (*strongly typed languages*)

Koncept slabih tipova podataka

- Informacija o tipu koristi se samo na mašinskom nivou da bi se odredio format i veličina memorijskog prostora potrebnog da se podatak registruje.
- Na nivou izvornog koda dozvoljava mešovite izraze pri čemu se vrši implicitna konverzija tipova podataka.
- Jezici sa slabom tipizacijom: C, C++

Koncept slabih tipova podataka - Primer

- Posmatrajmo prethodni primer:

```
double x;  
int i, j, k;  
i = x;  
k = x * j;
```

- Očekivana implicitna konverzija konverzija koja će se izvršiti u prvoj naredbi:

```
i = (int) x; //moguć gubitak informacije
```

- Moguće konverzije prilikom izvršenja druge naredbe:

```
k = ((int)x) * j;
```

```
k = (int) (x * ((double)j));
```

Koncept jakih tipova podataka

- Tip podataka određuju sledeći elementi:
 - Skup vrednosti
 - Format registrovanja podataka
 - Skup operacija koje se nad podacima mogu izvršavati
 - Skup funkcija za konverzije - uspostavljanje veza sa drugim tipovima podataka;
- Sve definicije tipa moraju da budu javne, eksplicitne. Nisu dozvoljene implicitne definicije tipova.
- Promenljivoj se definiše samo jedan tip.

Koncept jakih tipova podataka

- Dozvoljeno je dodeljivanje vrednosti samo odgovarajućeg tipa.
- Nad podacima su dozvoljene samo operacije obuhvaćene tipom kojem pripadaju.
- Tip je zatvoren u odnosu na skup operacija koji obuhvata. Te operacije mogu se primenjivati samo nad operandima tog tipa podataka. Mešoviti izrazi nisu dozvoljeni.
- Dodeljivanje vrednosti raznorodnih tipova i operacije nad raznorodnim operandima moguće je samo uz eksplicitnu konverziju tipova.
- Programski jezici sa jakom tipizacijom:
 - C#, Java, Python,...

Koncept jakih tipova podataka - primer

```
float x;  
int i;  
boolean b;  
char c;  
/* naredbe u kojima prevodilac  
   prijavljuje „type error” */  
i = 'A';  
x = i + 5.0;  
c = 10;  
i = i || 7;
```

Odstupanja od koncepta jakih tipova u programskim jezicima C# i Java

- Dozvoljena dodela izraza „nižeg“ numeričkog tipa promenljivoj „višeg“

```
double x;  
int i;  
i = x;      //GRESKA!  
x = i;      //KOREKTNO!
```

- Dozvoljeni mešoviti izrazi sa podacima numeričkog tipa. Uvek se vrši konverzija podatka „nižeg“ numeričkog tipa u „viši“ i operacija izvodi u „višem“ tipu:

$x * i \longleftrightarrow x * ((double)i)$

Odstupanja od koncepta jakih tipova u programskim jezicima C# i Java

- Dozvoljena operacija + nad stringovima i podacima bilo kog drugog tipa pri čemu se najpre vrši implicitna konverzija tog drugog podatka u string i nakon toga se operacija izvodi u tipu string:

```
string s;  
int i;  
s - i;      //GRESKA!  
s + i;      //KOREKTNO!
```

Ekvivaltnost tipova

Naredba:

```
id = <izraz>
```

može da se izvrši ukoliko su promenljiva sa leve strane i izraz na desnoj strani ekvivalentnog tipa (izuzimajući slučajeve kada se vrši implicitna konverzija).

Kada su 2 tipa ekvivalenta?

Ekvivalentnost tipova

- Strukturna ekvivalentnost
 - 2 tipa su ekvivalentna ako mogu da predstavljaju isti skup vrednosti
- Eksplicitna (imenovana ekvivalentnost)
 - 2 tipa su ekvivalentna ukoliko su identična
- C je jezik sa strukturnom ekvivalentnošću tipova:

```
typedef int dan;  
typedef int godina;  
dan d=10;  
godina g;  
g = d;
```
- U jezicima sa eksplicitnom ekvivalentnošću tipova ova dodela ne bi bila moguća bez eksplicitne konverzije tipova.