

Numerička analiza

2. predavanje – dodatak

Autor: Saša Singer

Predavač: Nela Bosner

nela@math.hr

web.math.hr/~nela/nad.html

PMF – Matematički odjel, Zagreb

Sadržaj predavanja

- Prikaz realnih brojeva — “floating-point” standard:
 - Osnovni oblik “floating-point” prikaza — mantisa i eksponent.
 - Greške zaokruživanja u prikazu.
 - Jedinična greška zaokruživanja.
 - IEEE standard — tipovi: *single*, *double*, *extended*.

Prikaz “realnih” brojeva u računalu — IEEE standard

Uvod u prikaz realnih brojeva

Kako pohraniti “jako velike” ili “jako male” brojeve?
Recimo (dekadski pisano):

67800000000.0 0.000002078

Koristimo tzv. **znanstvenu** notaciju u kojoj

- **prvo** pišemo **vodeće značajne znamenke** broja,
- a **zatim** pišemo **faktor** koji ima oblik **baza na odgovarajući eksponent**, tj. potenciju baze.

Uz dogovor da vodeći dio bude između 1 i 10 (strogo ispod),
to izgleda ovako:

$6.78 \cdot 10^{10}$ $2.078 \cdot 10^{-6}$.

Prikaz realnih brojeva

U računalu se binarni zapis realnog broja pohranjuje u znanstvenom formatu:

$$\text{broj} = \text{predznak} \cdot \text{mantisa} \cdot 2^{\text{eksponent}}.$$

Mantisa se uobičajeno (postoje iznimke!) pohranjuje u tzv. **normaliziranom** obliku, tj.

$$1 \leq \text{mantisa} < (10)_2.$$

I za pohranu **mantise** i za pohranu **eksponenta** rezervirano je **konačno** mnogo binarnih znamenki. Posljedice:

- prikaziv je samo neki **raspon** realnih brojeva,
- niti svi brojevi unutar prikazivog raspona **nisu prikazivi** (mantisa predugačka) \implies **zaokruživanje**.

Prikaz realnih brojeva (nastavak)

Primjer: Znanstveni prikaz binarnih brojeva:

$$1010.11 = 1.01011 \cdot 2^3$$

$$0.0001011011 = 1.01011 \cdot 2^{-4}$$

Primijetite da se vodeća jedinica u normaliziranom obliku **ne mora** pamtiti (ako je broj $\neq 0$).

- Taj bit se može upotrijebiti za pamćenje **dodatne znamenke mantise**.

Tada se vodeća jedinica zove **skriveni bit** (engl. hidden bit) — jer se **ne pamti**.

Ipak ovo je samo pojednostavljeni prikaz realnih brojeva.

Stvarni prikaz realnih brojeva

Najznačajnija promjena obzirom na pojednostavljeni prikaz:

- **eksponent** se prikazuje u “zamaskiranoj” ili “pomaknutoj” formi (engl. “biased form”).

To znači da se stvarnom **eksponentu**

- **dodaje konstanta** — takva da je “pomaknuti” eksponent uvijek **pozitivan**.

Ta konstanta ovisi o broju bitova za eksponent i bira se tako da je prikaziva

- **recipročna** vrijednost **najmanjeg pozitivnog normaliziranog broja**.

Takav “pomaknuti” eksponent naziva se **karakteristika**, a normaliziranu mantisu neki zovu i **signifikand**.

Oznake

Oznake:

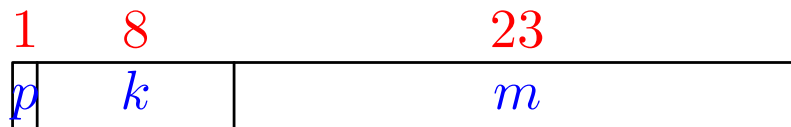
- **Crveno** — duljina odgovarajućeg polja (u bitovima), bitove brojimo od 0 zdesna nalijevo (kao i obično),
- p — predznak: 0 za pozitivan broj, 1 za negativan broj,
- k — karakteristika,
- m — mantisa (signifikand).
- Najznačajniji bit u odgovarajućem polju je najljeviji.
- Najmanje značajan bit u odgovarajućem polju je najdesniji.

Stvarni prikaz tipa single

“Najkraći” realni tip je tzv. realni broj **jednostruke** točnosti — u C-u poznat kao **float**.

On ima sljedeća svojstva:

- duljina: 4 byte-a (32 bita), podijeljen u tri polja.



- u mantisi se ne pamti vodeća jedinica ako je broj normaliziran,
- stvarni eksponent broja e , $e \in \{-126, \dots, 127\}$,
- karakteristika $k = e + 127$, tako da je $k \in \{1, \dots, 254\}$,
- karakteristike $k = 0$ i $k = 255$ koriste se za “posebna stanja”.

Stvarni prikaz tipa single (nastavak)

Primjer: Broj $(10.25)_{10}$ prikažite kao broj u jednostrukoj točnosti.

$$\begin{aligned}(10.25)_{10} &= \left(10 + \frac{1}{4}\right)_{10} = (10 + 2^{-2})_{10} \\ &= (1010.01)_2 = 1.01001 \cdot 2^3.\end{aligned}$$

Prema tome je:

$$p = 0$$

$$k = e + 127 = (130)_{10} = (2^7 + 2^1)_{10} = 1000\ 0010$$

$$m = 0100\ 1000\ 0000\ 0000\ 0000\ 000$$

Prikazi nule: $k = 0, m = 0$

Realni broj **nula** ima **dva** prikaza:

● mantisa i karakteristika su joj **nula**,
a predznak može biti

● **0** — “pozitivna nula”, ili

● **1** — “negativna nula”.

Ta dva prikaza nule su:

$$+0 = 0000\ 0000\ 0000\ 0000\ 0000\ 0000\ 0000\ 0000$$

$$-0 = 1000\ 0000\ 0000\ 0000\ 0000\ 0000\ 0000\ 0000$$

Smatra se da su vrijednosti ta dva broja **jednake** (kad se uspoređuju).

Denormalizirani brojevi: $k = 0, m \neq 0$

Ako je $k = 0$, a postoji **barem jedan** znak mantise koji **nije** nula, onda se kao eksponent uzima -126 . Mantisa takvog broja **nije normalizirana** i počinje s $0.m$.

Takvi brojevi zovu se **denormalizirani brojevi**.

Primjer: Kako izgleda prikaz realnog broja

$$0.000\ 0000\ 0000\ 0000\ 0000\ 1011 \cdot 2^{-126}?$$

Rješenje:

$$p = 0$$

$$k = 0000\ 0000$$

$$m = 000\ 0000\ 0000\ 0000\ 0000\ 1011$$

Plus i minus beskonačno: $k = 255, m = 0$

Ako je $k = 255$, a mantisa jednaka 0, onda

- $p = 0$ — prikaz $+\infty$, skraćena oznaka **+Inf**,
- $p = 1$ — prikaz $-\infty$, skraćena oznaka **-Inf**.

Primjer: Prikaz broja $+\infty$ ($-\infty$) je

$$p = 0 \quad (p = 1)$$

$$k = 1111 \ 1111$$

$$m = 000 \ 0000 \ 0000 \ 0000 \ 0000 \ 0000$$

Nije broj: $k = 255, m \neq 0$

Ako je $k = 255$ i postoji bar jedan bit mantise različit od nule, onda je to signal da se radi o pogrešci (recimo — dijeljenje nule s nulom, vađenje drugog korijena iz negativnog broja i sl.)

Tada se takva pogreška kodira znakom za Not a Number ili, skraćeno, s NaN.

Primjer.

$$p = 0$$

$$k = 1111\ 1111$$

$$m = 000\ 0000\ 0000\ 0101\ 0000\ 0000$$

Greške zaokruživanja

Postoje realni brojevi koje **ne možemo egzaktno** spremiti u računalo, čak i kad su **unutar** prikazivog raspona brojeva. Takvi brojevi imaju **predugačku mantisu**.

Primjer: Realni broj (u binarnom zapisu)

$$a = 1.0001\ 0000\ 1000\ 0011\ 1001\ 0111$$

ima **25** znamenki mantise i **ne može** se egzaktno spremiti u realni broj jednostruke preciznosti **float** u C-u, koji ima **23 + 1** znamenki za mantisu.

Procesor tada pronalazi **dva najbliža prikaziva** susjeda a_- , a_+ , broju a , takva da vrijedi

$$a_- < a < a_+.$$

Greške zaokruživanja (nastavak)

U našem primjeru je:

$$a = 1.0001\ 0000\ 1000\ 0011\ 1001\ 0111$$

$$a_- = 1.0001\ 0000\ 1000\ 0011\ 1001\ 011$$

$$a_+ = 1.0001\ 0000\ 1000\ 0011\ 1001\ 100$$

Nakon toga, **zaokružuje** se rezultat. Zaokruživanje može biti:

- prema **najbližem** broju (**standardno**, engl. **default**, za IA-32 procesore) — ako su dva susjeda **jednako** udaljena od a , izabire **parni** od ta dva broja (zadnji bit je 0),
- prema **dolje**, tj. prema $-\infty$,
- prema **gore**, tj. prema ∞ ,
- prema **nuli**, tj. odbacivanjem “viška” znamenki.

Greške zaokruživanja (nastavak)

Standardno zaokruživanje u našem primjeru:

$$a = 1.0001\ 0000\ 1000\ 0011\ 1001\ 0111$$

$$a_- = 1.0001\ 0000\ 1000\ 0011\ 1001\ 011$$

$$a_+ = 1.0001\ 0000\ 1000\ 0011\ 1001\ 100$$

Ovdje su a_- i a_+ jednako udaljeni od a , pa je zaokruženi a jednak a_+ , jer a_+ ima **parni** zadnji bit (jednak je 0).

Jedinična greška zaokruživanja

Ako je $x \in \mathbb{R}$ unutar raspona brojeva prikazivih u računalu, onda se, umjesto x , sprema zaokruženi prikazivi broj $fl(x)$.

Time smo napravili grešku zaokruživanja $\leq \frac{1}{2}$ “zadnjeg bita” mantise, i taj broj se zove

• jedinična greška zaokruživanja (engl. unit roundoff).

Standardna oznaka je u . Za float je

$$u = 2^{-24} \approx 5.96 \cdot 10^{-8}.$$

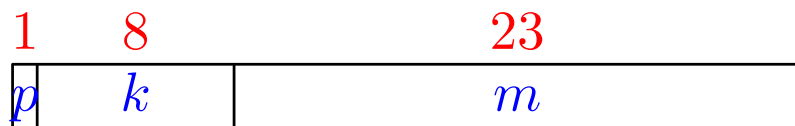
Vrijedi

$$fl(x) = (1 + \varepsilon)x, \quad |\varepsilon| \leq u,$$

gdje je ε relativna greška napravljena tim zaokruživanjem. Dakle, imamo vrlo malu relativnu grešku.

Prikaz brojeva jednostruke točnosti — sažetak

IEEE tip `single` = `float` u C-u:



Vrijednost broja je

$$v = \begin{cases} (-1)^p * 2^{(k-127)} * (1.m) & \text{ako je } 0 < k < 255, \\ (-1)^p * 2^{(-126)} * (0.m) & \text{ako je } k = 0 \text{ i } m \neq 0, \\ (-1)^p * 0 & \text{ako je } k = 0 \text{ i } m = 0, \\ (-1)^p * \text{Inf} & \text{ako je } k = 255 \text{ i } m = 0, \\ \text{NaN} & \text{ako je } k = 255 \text{ i } m \neq 0. \end{cases}$$

Raspon tipa float

Najveći prikazivi pozitivni broj je
 $\text{FLT_MAX} \approx 3.402823466 \cdot 10^{38}$, s prikazom

$$p = 0$$

$$k = 1111\ 1110$$

$$m = 111\ 1111\ 1111\ 1111\ 1111\ 1111$$

Najmanji prikazivi normalizirani pozitivni broj je
 $\text{FLT_MIN} \approx 1.175494351 \cdot 10^{-38}$, s prikazom

$$p = 0$$

$$k = 0000\ 0001$$

$$m = 000\ 0000\ 0000\ 0000\ 0000\ 0000$$

Raspon tipa float

Simboličke konstante `FLT_MAX`, `FLT_MIN` i još poneke vezane uz tip `float`, definirane su u datoteci zaglavlja `float.h` i mogu se koristiti u C programima.

Uočite:

- $1/\text{FLT_MIN}$ je **egzaktno** prikaziv (nađite prikaz),
- $1/\text{FLT_MAX}$ **nije egzaktno** prikaziv i zalazi u denormalizirane brojeve (tzv. “gradual underflow”).

Najmanji prikazivi denormalizirani pozitivni broj je $2^{-126} \cdot 2^{-23} = 2^{-149} \approx 1.4013 \cdot 10^{-45}$, s prikazom

$$p = 0$$

$$k = 0000\ 0000$$

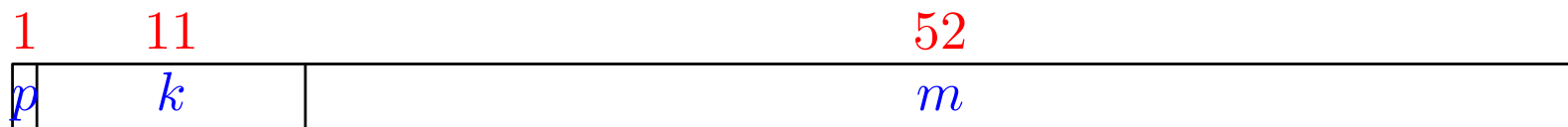
$$m = 000\ 0000\ 0000\ 0000\ 0000\ 0001$$

Stvarni prikaz tipa double

“Srednji” realni tip je tzv. realni broj **dvostruke** točnosti — u C-u poznat kao **double**.

On ima sljedeća svojstva:

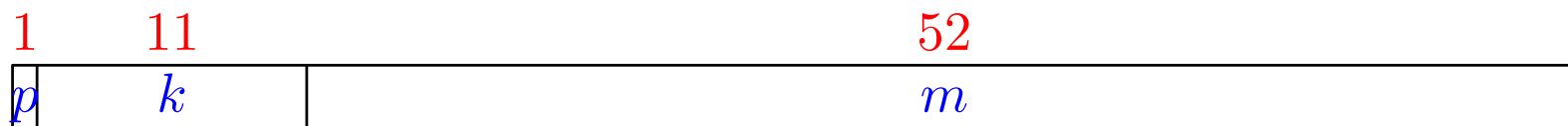
- Duljina: 8 byte-a (64 bita), podijeljen u **tri** polja.



- u mantisi se ne pamti vodeća jedinica ako je broj normaliziran,
- **stvarni eksponent** broja e , $e \in \{-1022, \dots, 1023\}$,
- **karakteristika** $k = e + 1023$, tako da je $k \in \{1, \dots, 2046\}$,
- **karakteristike** $k = 0$ i $k = 2047$ — “posebna stanja”.

Prikaz brojeva dvostruke točnosti — sažetak

IEEE tip `double` = `double` u C-u:



Vrijednost broja je

$$v = \begin{cases} (-1)^p * 2^{(k-1023)} * (1.m) & \text{ako je } 0 < k < 2047, \\ (-1)^p * 2^{(-1022)} * (0.m) & \text{ako je } k = 0 \text{ i } m \neq 0, \\ (-1)^p * 0 & \text{ako je } k = 0 \text{ i } m = 0, \\ (-1)^p * \text{Inf} & \text{ako je } k = 2047 \text{ i } m = 0, \\ \text{NaN} & \text{ako je } k = 2047 \text{ i } m \neq 0. \end{cases}$$

Jedinična greška i raspon tipa double

Jedinična greška zaokruživanja za `double` je

$$u = 2^{-53} \approx 1.11 \cdot 10^{-16}.$$

Broj $1 + 2u$ je **najmanji** prikazivi broj strogo **veći** od 1. Postoji

$$\text{DBL_EPSILON} = 2u \approx 2.2204460492503131 \cdot 10^{-16}.$$

Najveći prikazivi pozitivni broj je

$$\text{DBL_MAX} \approx 1.7976931348623158 \cdot 10^{308}.$$

Najmanji prikazivi normalizirani pozitivni broj je

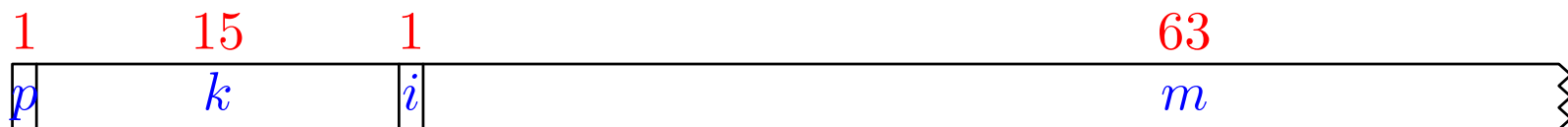
$$\text{DBL_MIN} \approx 2.2250738585072014 \cdot 10^{-308}.$$

Tip extended

Stvarno računanje (na IA-32) se obično radi u “proširenoj” točnosti — u C-u možda dohvatljiv kao `long double`.

On ima sljedeća svojstva:

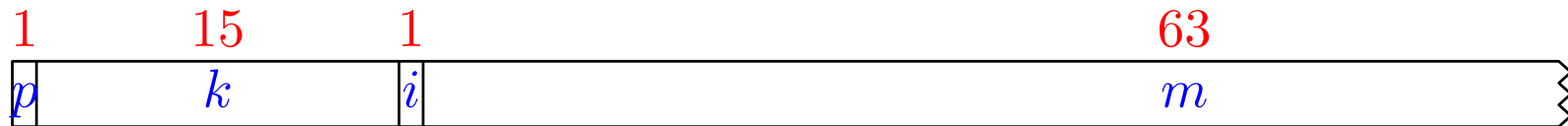
- Duljina: 10 byte-a (80 bita), podijeljen u četiri polja.



- u mantisi se pamti vodeći bit i mantise,
- stvarni eksponent broja e , $e \in \{-16382, \dots, 16383\}$,
- karakteristika $k = e + 16383$, tako da je $k \in \{1, \dots, 32766\}$,
- karakteristike $k = 0$ i $k = 32767$ — “posebna stanja”.

Prikaz brojeva proširene točnosti — sažetak

IEEE tip *extended*:



Vrijednost broja je

$$v = \begin{cases} (-1)^p * 2^{(k-16383)} * (i.m) & \text{ako je } 0 \leq k < 32767, \\ (-1)^p * \text{Inf} & \text{ako je } k = 32767 \text{ i } m = 0, \\ \text{NaN} & \text{ako je } k = 32767 \text{ i } m \neq 0. \end{cases}$$

Uočite da **prva** mogućnost uključuje:

• $+0$, -0 i **denormalizirane** brojeve (za $k = 0$),

jer se **pamti** vodeći “cjelobrojni” bit i mantise.