Programovanie

3. ročník – odborná prax

Ing. Daniela Kravcová

Základné funkcie

```
# include <stdlib.h>          //volanie zabudovanej knižnice ....
# include <stdio.h>           //deklarácia funkcie v/v
# include <conio.h>           //umožňuje deklarovať funkciu clrscr()

main()                        // hlavný program
    {                         Za main () idú  kučeravé zátvorky { }.
                              V kučeravých zátvorkách je telo funkcie.
    funkcia                   // funkcia, kde program začína,
                                 a keď táto dobehne, program skončí.
    }
```

PARAMETRE VSTUPNÉHO RIADKU

Typy premenných

Definícia typu premenných môže obsahovať:

1. **základný preddefinovaný typ** (char, int, float, double)
2. **odvodený typ** (smerník, pole, struct, union)
3. **vymenovaný typ** (enum)
4. **užívateľom definovaný typ** (typedef)
5. **prázdny typ** (void)

Základný preddefinovaný typ

Základné typy môžu byť modifikované klasifikátorom:

Typy pre ukladanie **celé čísla bez znamienka**:

- **unsigned char -** 8 – bitové bez znamienkové číslo,
- **unsigned short -** 16 – bitové bez znamienkové číslo,
- **unsigned long -** 32 – bitové bez znamienkové číslo

Typy pre ukladanie **celé čísla so znamienkom**:

- **signed char -** 16 – bitové celé číslo,
- **short -** 32 – bitové celé číslo,
- **long -** 64 – bitové celé číslo,
- **float -** 32 – bitové reálne číslo,
- **double -** 64 – bitové reálne číslo

Formátové špecifikácie pre printf a scanf

```
scanf("%d",&i);
                    // % ... ukazovač (pointer) - ukazuje umiestnenie premennej
                    // &r - premenná, ktorej je daná hodnota - priradzovač
```

```
printf("%d je parne cislo", i);
```

%d – vypísať alebo vytlačiť hodnotu typu int

%f – typu float

%lf – typu double

%c – typu char

%s – typu string

%x – číslo typu int v šestnástkovej sústave

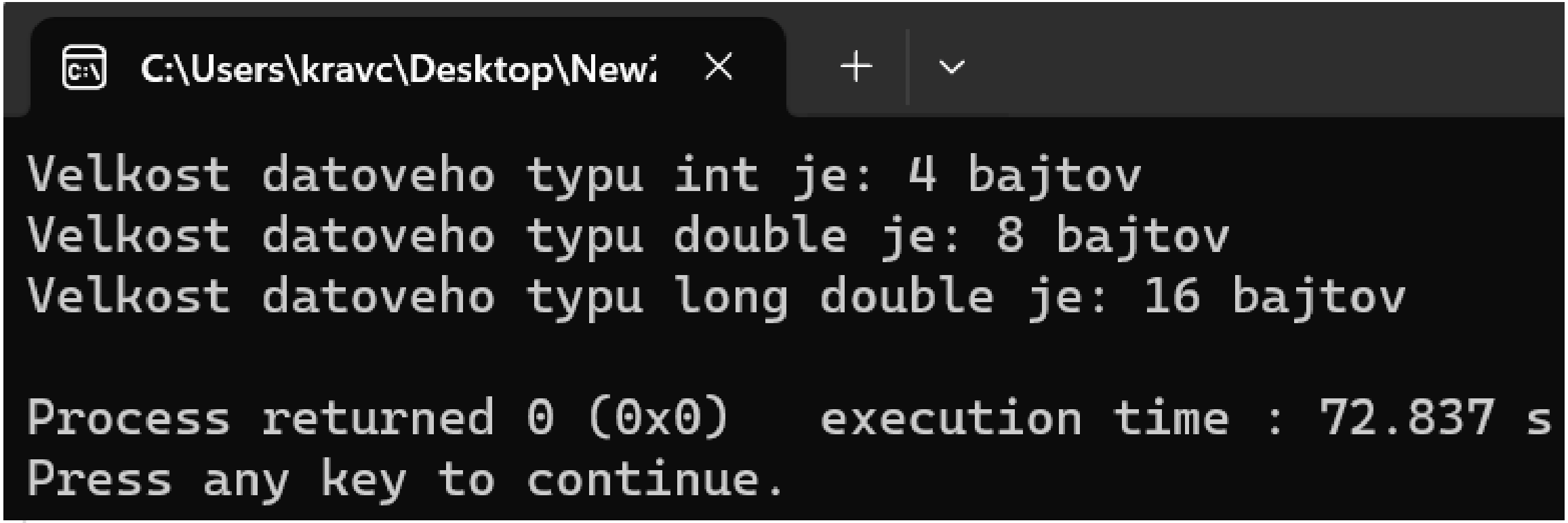
%o – číslo int v osmičkovej sústave

Základný preddefinovaný typ

Príklad1:

pomocou operátora sizeof() zistite veľkosť dátových typov int, double, long double, float, ...

Výstup:

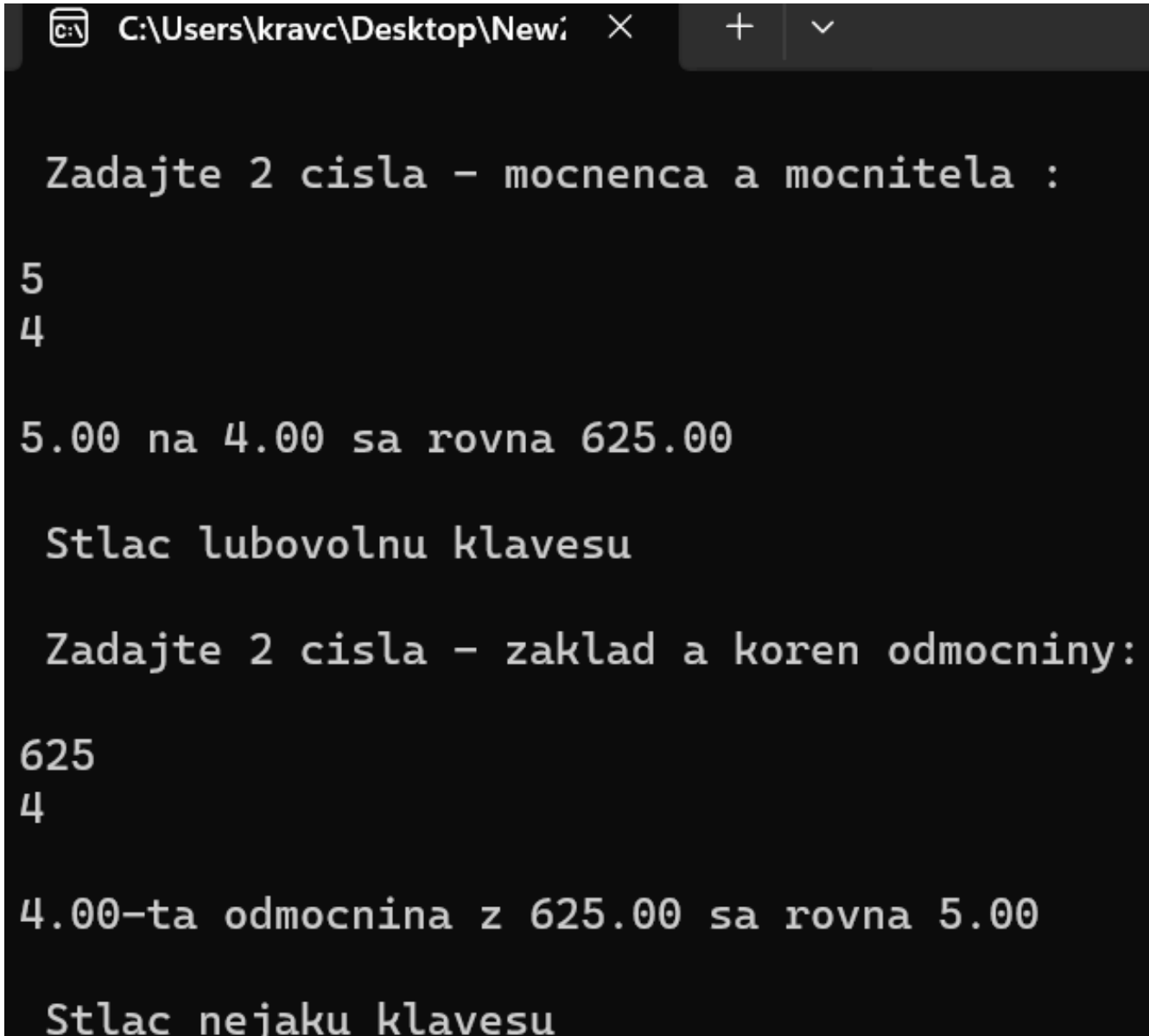


Príklad2:

Napíšte program pre výpočet mocniny a odmocniny dvoch racionálnych čísel.

(Použite funkciu **pow** definovanú v knižnici <math.h>)

Výstup:



Parametre vstupného riadku

- Funkciám sa niekedy hovorí podprogramy alebo **subrutiny**.
- Ak funkcia nevracia žiadnu hodnotu, môže sa jej v niektorých jazykoch hovoriť **procedúra.**
- U väčších aplikácií, ktoré majú mnoho funkcií, sa funkcie združujú do tzv. **Modulov.**

napr. v podobe **#include <stdio.h>**, ktorým načítame knižnicu (*modul*) pre prácu so štandardným vstupom a výstupom (teda pre nás s konzolou).

Parametre: **int argc, char *argv[]**

int argc (počet argumentov):

argc - je celočíselná premenná, ktorá udáva počet argumentov príkazového riadku, ktorý program obdržal pri spustení.
- táto hodnota bude minimálne 1, pretože vždy existuje aspoň jeden argument, a to je názov spusteného programu.

char *argv[] (vektor argumentov):

argv - je pole reťazcov (položiek), kde každý reťazec predstavuje jeden argument príkazového riadku.
- prvý prvok (argv[0]) obsahuje názov spusteného programu.
- ostatné prvky obsahujú ďalšie argumenty príkazového riadku.

int main(int argc, char *argv[])

```
int main(int argc, char *argv[])                    // argc obsahuje počet argumentov
    {                                               // argv je pole reťazcov s argumentmi
    printf("Pocet argumentov: %d\n", argc);         // Vypíšeme všetky argumenty
        for (int i = 0; i < argc; i++)
            {
                printf("Argument %d: %s\n", i, argv[i]);
            }
    return 0;
    }
```

Operátor pretypovanie

Pomocou **pretypovania** môžeme zmeniť typ existujúcich údajových typov. Vďaka tomuto môžeme prepísať automatickú konverziu.

Typy pretypovania:

- **implicitné –** prebieha automaticky
- **explicitné –** prebieha obráteným smerom

Implicitné pretypovanie:

- **implicitné -** ak sa pôvodný dátový typ „vojde" do dátového typu cieľového objektu alebo prevodom nedôjde k strate informácie.

```
int a = 3;
double b = 2.123;
double c;
c = a + b;                    // a sa prevedie na double hodnotu 3.000
```

Tabuľka priorít (od najnižšieho k najvyššiemu)

Int
unsigned int
long
unsigned long
float
double
long double

Explicitné pretypovanie:

- **explicitné –** vynútené pretypovanie sa použije tam, kde nie je implicitný prevod možný.
  - nastáva tu problém so zmysluplnosťou týchto prevodov
  - zároveň je potrebné dávať pozor na stratu informácií, ku ktorým pri tejto konverzií dochádza.

```
double a = 5.3;
int b;
b = (int)a;          // do premennej b sa uloží orezané a, tj. 5
```

Pretypovanie v C/C++ sa vykonáva tak, že sa pred výraz napíše do zátvorky typ, na ktorý chceme pretypovať.

```
int a = 60, b = 7;
float c, d;
char ch = 'x‘;


c = a / b;
d = (float) a / b;
printf("Celociselny vysledok:  %+4.3f\n", c);
printf("Racionalne delenie:  %+4.3f\n", d);
printf("ASCII kod znaku %c je %d\n", ch, (int) ch);
```

Skúste modifikácie !!!