

Jednorozměrná pole

5. cvičení

Jiří Zacpal

KMI/ZP1 – Základy programování 1

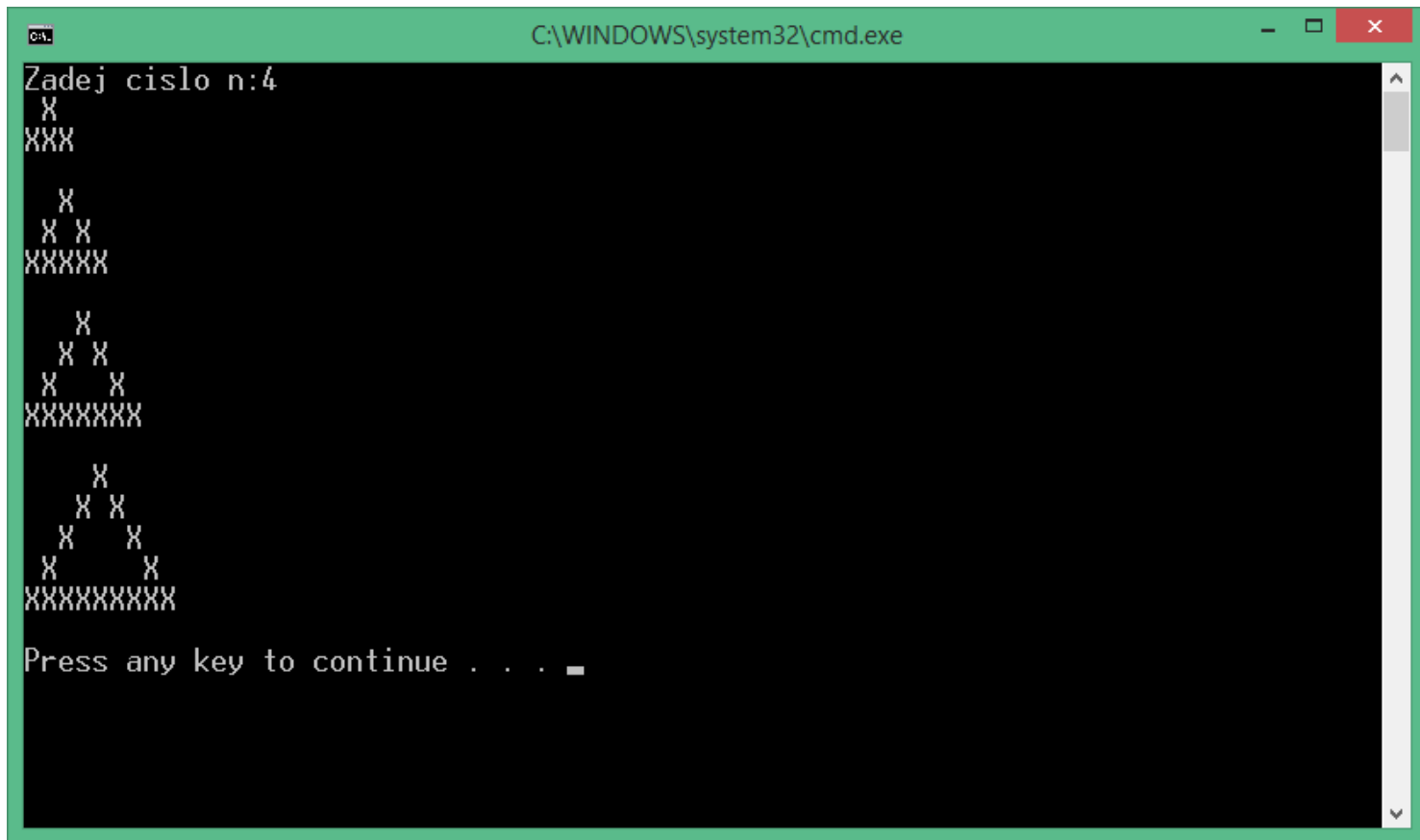
1. Písemná práce

- Příští hodinu
- Rozdělení na dvě poloviny
- 35 minut na řešení příkladu

Zadání 1. příkladu

Napište program, který pro zadané číslo n postupně vytiskne na obrazovku trojúhelníky, které budou mít $i+1$ řádků a $2*i+1$ sloupců ($i=1, \dots, n$). Maximální hodnota n bude 11 (součástí programu musí být i test přípustnosti n).

Zadání 1. příkladu



A screenshot of a Windows command prompt window titled "C:\WINDOWS\system32\cmd.exe". The window has a black background with white text. The text displayed is as follows:

```
Zadej cislo n:4
X
XXX

  X
 X X
XXXXX

    X
  X X
 X  X
XXXXXXXX

      X
    X X
  X  X
 X   X
XXXXXXXXXX

        X
      X X
    X  X
  X   X
 XXXXXXXXXX

Press any key to continue . . .
```

The output shows four patterns of 'X' characters, each corresponding to a value of 'n' from 1 to 4. Each pattern consists of a top row of 'X's and a bottom row of 'X's, with the number of 'X's in each row increasing by 2 for each subsequent 'n'. The patterns are centered horizontally. The prompt "Press any key to continue . . ." is at the bottom.

Deklarace pole

- syntaxe bez inicializace:
`typ identifikátor[velikost];`
- příklad:
`int moje_cisla[10];`
- syntaxe s inicializací:
`typ indent[vel]={prvek_1, ..., prvek_N};`
- příklady:
`int cisla[5]={1,2,3,4,5};`
`int cisla[]={1,2,3,4,5};`
`char str[]="Ahoj svete";`
- přístup k prvkům pole pomocí operátoru indexu `[]`

Příklad 1

```
main()
{
    int i,soucet;
    int pole[10];
    for (i=0; i<10; i++)
    {
        pole[i]= i+1;
    }
    printf("Pole obsahuje cisla: ");
    for (i=0; i<10; i++)
    {
        printf("%i, ", pole[i]);
    }
    soucet=0;
    for (i=0; i<10; i++)
    {
        soucet+=pole[i];
    }
    printf("\nPrumer prvku pole je:
    %.2f\n", (float)soucet/10);
}
```

1	pole[0]
2	pole[1]
3	pole[2]
4	pole[3]
5	pole[4]
6	pole[5]
7	pole[6]
8	pole[7]
9	pole[8]
10	pole[9]

Textové řetězce

- pole znaků ukončených znakem ' \0 '
- velikost pole je délka řetězce +1
- funkce pro práci s řetězcí jsou ve knihovně `string.h`

Funkce pro práce s řetězcí 1/3

- `char *strcat(char *dest, const char *src);`
 - Funkce připojí řetězec `src` k řetězci `dest` a vrátí ukazatel na řetězec `dest`.
- `char *strncat(char *dest, const char *src, size_t n);`
 - Jako funkce `strcat`, ale přidá jen `n` znaků z `src`.
- `int strcmp(const char *s1, const char *s2);`
 - Porovnává řetězce `s1` a `s2`. Pokud je `s1` menší než `s2`, pak vrátí záporné číslo, pokud si jsou řetězce rovny, pak vrátí nulu, a pokud je `s1` větší než `s2`, pak vrátí kladné číslo.
- `int strncmp(const char *s1, const char *s2, size_t n);`
 - Jako `strcmp`, porovnává však jen prvních `n` znaků.

Funkce pro práce s řetězcí 2/3

- `char *strcpy(char *dest, const char *src);`
 - Zkopíruje znaky řetězce `src` do řetězce `dest` a vrátí ukazatel na `dest`.
- `char *strncpy(char *dest, const char *src, size_t n);`
 - Jako `strcpy`, ale kopíruje maximálně `n` znaků. (Je-li jich právně `n`, nepřidá zarážku `'\0'`.)
- `size_t strlen(const char *s);`
 - Vrací délku řetězce `s`.
- `char *strchr(const char *s, int c);`
 - Vrací ukazatel na první pozici v řetězci `s`, na níž se vyskytuje znak `c`, nebo `NULL`, pokud znak `c` nenajde.
- `char *strrchr(const char *s, int c);`
 - Jako `strchr`, ale hledá první výskyt zprava.

Funkce pro práce s řetězcí 3/3

- `char *strstr(const char *haystack, const char *needle);`
 - Vrací ukazatel na první výskyt řetězce `needle` v řetězci `haystack` nebo `NULL`, pokud jej nenajde.
- poznámky k deklaracím výše:
 - Typ `size_t` je ekvivalent typu `unsigned int`, definovaný v knihovně `string.h`.
 - Typ před názvem funkce určuje typ návratové hodnoty, v závorce za názvem funkce jsou pak typy a jména jednotlivých vstupních parametrů (viz příští seminář).

Příklad 2

```
main()
{
    char *mezera, celejmeno[40], jmeno[15], prijmeni[20];
    int i;

    printf("Zadej jmeno:");
    i=0;
    do
    {
        celejmeno[i++]=getchar();
    }
    while(celejmeno[i-1]!='\n');
    celejmeno[i-1]='\0';

    mezera=strchr(celejmeno, ' ');
    i=1;
    while(mezera[i]!='\0')
    {
        prijmeni[i-1]=mezera[i];
        i++;
    }
    prijmeni[i-1]='\0';

    i=0;
    while(celejmeno[i]!=' ') i++;
    strncpy(jmeno, celejmeno, i);
    jmeno[i]='\0';

    printf("Cele jmeno: %s\n", celejmeno);
    printf("Jmeno: %s\n", jmeno);
    printf("Prijmeni: %s\n", prijmeni);
}
```

Příklad 2

```
main_2b()
{
    char *mezera, celejmeno[40], jmeno[15], prijmeni[20];
    int i, j;

    printf("Zadej jmeno:");
    gets(celejmeno);

    i=0;
    while(celejmeno[i]!=' ')
    {
        jmeno[i]=celejmeno[i];
        i++;
    }
    jmeno[i++]='\0';
    j=0;
    while(celejmeno[i]!='\0')
    {
        prijmeni[j]=celejmeno[i];
        i++;
        j++;
    }
    prijmeni[j]='\0';

    printf("Cele jmeno: %s\n", celejmeno);
    printf("Jmeno: %s\n", jmeno);
    printf("Prijmeni: %s\n", prijmeni);
}
```