KATEDRA INFORMATIKY
PŘÍRODOVĚDECKÁ FAKULTA
UNIVERZITA PALACKÉHO V OLOMOUCI

# Dynamická práce s pamětí

## 9. cvičení

Jiří Zacpal

KMI/ZP1 – Základy programování 1

# Přetypování ukazatelů

- ukazatele jsou při deklaraci vztaženy ke konkrétnímu datovému typu
- lze však definovat také ukazatel (ukazatel na typ void), který není vázán na konkrétní datový typ
- s takto definovaným ukazatelem lze některé operace provádět pouze po provedení explicitní konverze
- příklad:

```
void *moje_pamet;
…
((int *)moje_pamet)++;
```

# Funkce pro práci s pamětí 1/2

- níže uvedené funkce jsou definovány v `stdlib.h`
- při výpočtu velikosti bloku se používá operátor `sizeof`
- dynamická alokace paměti umožňuje spravovat paměť zcela libovolně (např. data mohou existovat nezávisle na lokálních proměnných bloku, ve kterém byly vypočteny)
- pokud není možné paměť alokovat, vrací funkce `NULL`
- `void *malloc(size_t size);`
  alokuje paměť velikosti size bytů a vrací ukazatel na přidělenou oblast paměti
- `void *calloc(size_t items, size_t size);`
  alokuje paměť pro items prvků o velikosti size bytů (items*size bytů), navíc tuto paměť vyplní nulami

# Funkce pro práci s pamětí 2/2

- `void *relloc(void* old, size_t size);`
  změna velikosti dříve alokovaného bloku paměti old na velikost size bytů
- `void free(void* block);`
  uvolnění paměti block; funkce by se měla zavolat vždy, když víme, že už daný blok paměti nebudeme v programu používat

# Příklad 1

```c
#include<stdio.h>
#include<stdlib.h>

int main()
{
     int i,j, *p;
     char r;
     printf("Kolik chcete nacist cisel?");
     scanf("%d",&i);
     for(j=0;j<i;j++)
     {
          if(j==0)
             p=(int*)malloc(sizeof(int));
          else
             p=(int*)realloc(p,sizeof(int)*(j+1));
          if(p==NULL)return 0;
          printf("Zadejte cislo:");
          //scanf("%i",(p+j));
          scanf("%i",&p[j]);
     }
     printf("Zadana cisla: ");
     for(j=0;j<i;j++) printf("%d, ",p[j]);
     free(p);
     return 1;
}
```