Aula 12 - Ponteiros de função

Assuntos

1. Ponteiros de função

Ponteiro de função

- Um ponteiro aponta para um endereço de memória
- Uma função ocupa um espaço de memória
- Então podemos ter um ponteiro de função
- E podemos combinar com o `typedef` para definir este tipo de função

Ponteiro de função

```c
#include <stdio.h>

void minha_funcao(){
  printf("chamada da minha funcao\n");
}

int main(){
  typedef (*funcao)();
  funcao f = minha_funcao;
  f();
}
```

Ponteiro de função - parte 1

```c
#include <stdio.h>

int soma(int x, int y){
  printf("soma\n");
  return x + y;
}

int subtracao(int x, int y){
  printf("subtracao\n");
  return x - y;
}
```

Ponteiro de função - parte 2

```
int divisao(int x, int y){
  printf("divisao\n");
  return x / y;
}

int multiplicacao(int x, int y){
  printf("multiplicacao\n");
  return x * y;
}
```

Ponteiro de função - parte 3

```
int main(){

  typedef int (*def)(int, int);

  def fs[] = {soma, subtracao, divisao, multiplicacao};

  int len = 4;
  int res[len];
  int x = 1;
  int y = 2;

  for(int i=0;i<len;i++)
    res[i] = fs[i](x, y);

  for(int i=0;i<len;i++)
    printf("%d ", res[i]);
}
```

Ponteiro de função

- Podemos usar para escolher qual função será executada
- Pode facilitar o uso de `if...else if...else` e `switch`
- Apesar de parecido não é o mesmo conceito de *first class citzen* de outras linguagens, como Python

`void`

- O tipo `void` em C é usado para descrever funções e variáveis sem tipos
- Para um ponteiro, isto pode ser interesssante pois se não possui um tipo específico, este ponteiro pode apontar para qualquer tipo

void

```c
#include <stdio.h>

int main(){
  void *p;
  int i = 1;
  float f = 3.14;

  p = &i;
  printf("*p = %d\n", *(int *)p);
  p = &f;
  printf("*p = %f\n", *(float *)p);
}
```