Exercícios
Subrotinas em “C”

Prof. Sérgio Luis Cechin

Exercício 1

- Escrever uma subrotina para implementar a função “*uint2str*(**char** **s*, **uint** *n*)”
- A função converte o valor em “*n*” para um string em “*s*”
- **uint**: inteiro sem sinal de 16 bits
- **char**: formato “char” do “C”
  - Seqüência de caracteres ASCII terminada pelo caractere “nul” (0x00 ou ‘\0’)

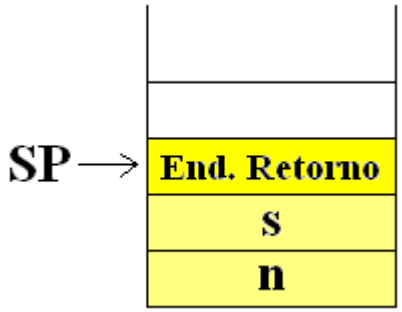
Solução

```
WORD        Tab10 = { 10000, 1000, 100, 10 };
void uint2str (char *s, uint n) {
    int Lacos;
    int dig;
    WORD *pTab10;

    pTab10 = Tab10;
    Lacos = 4;
    do {
            dig = '0';
            while (n >= *pTab10) {
                    ++dig;
                    n -= *pTab10;
            }
            *s++ = dig;
            ++pTab10;
    } while (--Lacos);
    *s++ = n + '0';
    *s = '\0';
}
```

Convenção (do compilador)
Parâmetros passados na pilha

Convenção (do compilador)
Colocados na pilha
da direita para a esquerda



1.uint2str.ced

Exercício 2

- Escrever uma subrotina para implementar a função "*int2str*(**char** *s, **int** *n*)"
- A função converte o valor em "*n*" para um string em "*s*"
- **int**: inteiro com sinal de 16 bits
- **char**: formato "char" do "C"
  - Seqüência de caracteres ASCII terminada pelo caractere "nul" (0x00 ou '\0')

Solução

```
void int2str (char *s, uint n) {
    if ( n < 0 ) {
            n = - n;
            *s++ = '-';
    }
    uint2str(s,n);
}
```

2.int2str.ced

Exercício 3

- Escrever uma subrotina para implementar a função "*str2int*(**int** **n*, **char** **s*)"
- A função converte o valor em "*s*" para um inteiro em "*n*"
- **int**: inteiro com sinal de 16 bits
- **char**: formato "char" do "C"
  - Seqüência de caracteres ASCII terminada pelo caractere "nul" (0x00 ou '\0')

Solução

```
// Versão com 2 parâmetros
void str2int (int *n, char *s) {
    *n = str2int(s);
}
```

Solução

```
// Versão com 1 parâmetro
int str2int (char *s) {
    int neg;
    int n;

    neg = false;
    if ( *s == '-' ) {
            neg = true;
            ++s;
    }
    n = 0;
    while ( *s != '\0' ) {
            n = 10 * n + (*s - '0');
            ++s;
    }
    if (neg) {
            n = -n;
    }
    return n;
}
```

Convenção (do compilador)
O retorno é sempre em R0

3.str2int.ced