rascunho - laboratório 10

estrutura das informações

- val 1 → nº inteiro
- val 2 → nº inteiro
- node → 2 nºs inteiros + aponta para o endereço do próximo nó

funções a serem definidas

- void puts (const char *str)
  - grava a string str na saída padrão e adiciona caracter de nova linha.
  - chegou em "\0" → grava "\n"
  - adiciona "\0" depois de "\n"
  - escreve

```
    li t2, 0              → contador
    mv t0, a0             → copia endereço (NÃO PRECISA)
checa_fim:
    lbu t1, 0(t0)         → analisa caracter
    beq zero, t1, fim     → encontrou "\0"
    addi t0, t0, 1        → próximo caractere
    addi t2, t2, 1        → contador
    pula checa_fim
fim:
    addi t2, t2, 1        → adiciona mais 1 caracter
    li t3, "\n"           → pula linha
    sw t3, 0(t0)          → troca "\0" por "\n"
    sw zero, 1(t0)        → escreve "\0" no final
    sub t0, t0, t2        → aponta para o início
escreve:
    li a0, 1              → stdout
    la a1, buffer_saida   → endereço de saída
    mv a2, t2             → nº de bytes
    li a7, 64             → código do sistema
    ecall                 → chama sistema
    ret
```

- char* gets (char *str)
  - recebe como entrada o ~~bu~~ buffer a ser preenchido.
  - lê os caracteres do stdin e os armazena como string e os armazena.
  - se chegou em "\n" → fim
  - se chegou em zero → fim
  - não precisa lidar com erros ou com o fim do arquivo

```
    li t1, 0              → contador
    li t2, "\n"           → pula linha
converte_entrada:
    lbu t3, 0(a0)         → 1º caracter
    addi t1, t1, 1        → contador + 1
    beq t3, t2, fim       → achou "\n"
    beq t3, zero, fim     → achou "\0"
    addi a0, a0, 1        → próximo caracter
    pula converte_entrada
fim:
    sw zero, 0(a0)        → troca "\n" por "\0"
    sub a0, a0, t1        → volta ao início
    ret                   → str na a0
```

Data: . . S T Q Q S S D

resumão - laboratório 10

funções a serem definidas

- int atoi (const char *str)
  - descarta caracteres de espaço e em branco
  - sinal opcional (+ ou -)
  - depois que a números acabam, fim

```
li t0,0          → soma
li t1,0          → contador nº lidos
li t2,'-'        → sinal -
li t3,'+'        → sinal +
li t4,10         → multiplicador
comeco_int:
lbu t5,0(a0)     → lê caracter
beq t5,t2,negativo → negativo
beq t5,t3,pegue  → positivo
li t6,'0'
blt t5,t6,nao_nº → não é número
li t6,'9'
bltt t6,t5,nao_nº → não é número
é_numero:
li t6,10
mul t0,t0,t6     → mult por 10
addi t5,t5,-'0'  → converte nº
add t0,t0,t5
addi t1,t1,1     → contador
pegue: addi a0,a0,1 → próximo caracter
jal comeco_int
nao_numero:
bgez t1,pegue
mul a0,t0,t4     → nº final
ret
negativo:
li t4,-1
jal pegue
```

- exit:
  li a7,93
  ecall

S T Q Q S S D Data: . .

recursão - laboratório 10

funções a serem definidas

- limited_list_search (node *node, int val)
  a_0 a_1

```
limited_list_search:
  lw  t1, 0(a0)
  lw  t2, 4(a0)

  add  t3, t1, t2

  beq  t3, a1, nomaiencontrada

  lw  t4, 8(a0)
  beqz  t4, fimlista

  addi  t5, t5, 1
  mv  a0, t4
  j  limited_list_search
nomaiencontrada:
  mv  a0, t5
  ret
fimlista:
  li  a0, -1
  ret
```