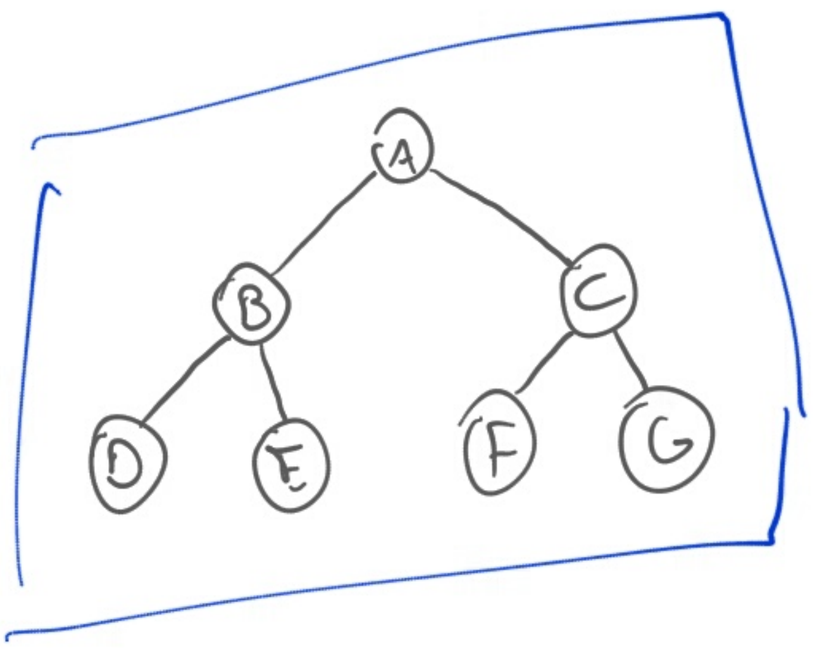
Atividade AULA - Daniel moraes - 180112333

main.asm | main.asm | main.asm | main.asm | main.asm | main.asm | *main.asm | main.asm | main.asm | »4

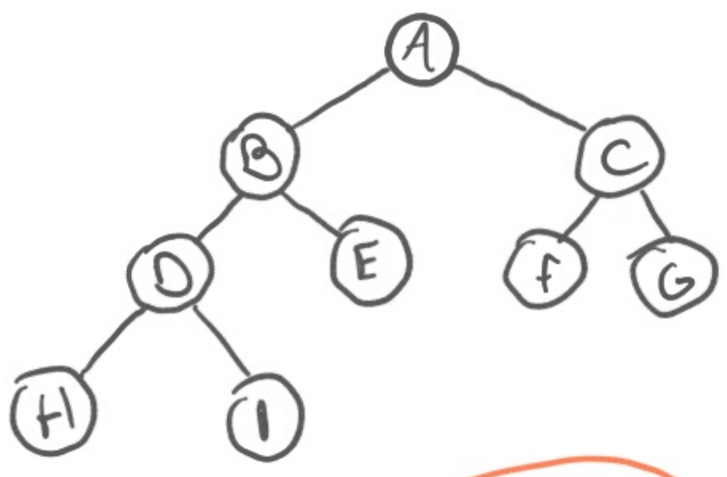
```
;-------------------------------------------------------------------------------
; MSP430 Assembler Code Template for use with TI Code Composer Studio
;
;
;-------------------------------------------------------------------------------
            .cdecls C,LIST,"msp430.h"       ; Include device header file

;-------------------------------------------------------------------------------
            .def    RESET                   ; Export program entry-point to
                                            ; make it known to linker.
;-------------------------------------------------------------------------------
            .text                           ; Assemble into program memory.
            .retain                         ; Override ELF conditional linking
                                            ; and retain current section.
            .retainrefs                     ; And retain any sections that have
                                            ; references to current section.

;-------------------------------------------------------------------------------
RESET       mov.w   #__STACK_END,SP         ; Initialize stackpointer
StopWDT     mov.w   #WDTPW|WDTHOLD,&WDTCTL  ; Stop watchdog timer


;-------------------------------------------------------------------------------
;Main loop here
;Daniel Moraes da Silva - Engenharia de Computação
;Percurso Arvore: Pré-Ordem
;-------------------------------------------------------------------------------
PO:         mov.b       #arv,R5
            mov.b       #ARV,R9
            mov.b       #0,R6               ;inidice inicia com a Raiz
            mov.b       #3,R7               ;indice da subarvore da direira
            mov.b       #6,R8               ;indice da subarvore da esquerda

NOD:        add.b       #1,R6               ;andando um indice do vetor
            mov.b       arv(R6),ARV(R6)     ;conferindo o atual indice colocando o elemento em R4
            cmp.b       R7,R6               ;comparar indices
            jhs         NOE                 ;se o atual indice for maior que subarvore da Direita (R7), Ir para subarvore Esquerda
            call        #NOD                ;se não continua percorrendo a subarvore da Direita

NOE:        add.b       #1,R6               ;andando um indice do vetor
            mov.b       arv(R6),ARV(R6)         ;conferindo o atual indice colocando o elemento em R4
            cmp.b       R8,R6               ;comparar indices
            jhs         FIM                 ;se o atual indice for maior que subarvore da Esquerda (R8), já percorreu toda a Arvore, chama F
            call        #NOE                ;se não continua percorrendo a subarvore da Direita


FIM:        jmp         $


            .data
arv:        .byte       "ABCDEFG"

            .data
ARV:        .byte       "0000000"

;-------------------------------------------------------------------------------
; Stack Pointer definition
;-------------------------------------------------------------------------------
            .global __STACK_END
            .sect   .stack

;-------------------------------------------------------------------------------
; Interrupt Vectors
;-------------------------------------------------------------------------------
            .sect   ".reset"                ; MSP430 RESET Vector
            .short  RESET


```

Pré - ordem



```
;-------------------------------------------------------------------------------
; MSP430 Assembler Code Template for use with TI Code Composer Studio
;
;
;-------------------------------------------------------------------------------
            .cdecls C,LIST,"msp430.h"       ; Include device header file

;-------------------------------------------------------------------------------
            .def    RESET                   ; Export program entry-point to
                                            ; make it known to linker.
;-------------------------------------------------------------------------------
            .text                           ; Assemble into program memory.
            .retain                         ; Override ELF conditional linking
                                            ; and retain current section.
            .retainrefs                     ; And retain any sections that have
                                            ; references to current section.

;-------------------------------------------------------------------------------
RESET       mov.w   #__STACK_END,SP         ; Initialize stackpointer
StopWDT     mov.w   #WDTPW|WDTHOLD,&WDTCTL  ; Stop watchdog timer


;-------------------------------------------------------------------------------
;Main loop here
;Daniel Moraes da Silva - Engenharia de Computação
;Percurso Arvore: Ordem Simetrica
;-------------------------------------------------------------------------------
OS:         mov.b       #arv,R4
            mov.b       #ARV,R5
            mov.b       #0,R6

SUBESQ:     mov.b       #7,R7               ;|
            mov.b       arv(R7),ARV(R6)     ;|> Imprime H
            add.b       #1,R6               ;|

            mov.b       #3,R7               ;|
            mov.b       arv(R7),ARV(R6)     ;|> Imprime D
            add.b       #1,R6               ;|

            mov.b       #8,R7               ;|
            mov.b       arv(R7),ARV(R6)     ;|> Imprime I
            add.b       #1,R6               ;|

            mov.b       #1,R7
            mov.b       arv(R7),ARV(R6)     ;|> Imprime B
            add.b       #1,R6

            mov.b       #4,R7
            mov.b       arv(R7),ARV(R6)     ;|> Imprime E
            add.b       #1,R6

            mov.b       #0,R7
            mov.b       arv(R7),ARV(R6)     ;|> Imprime A
            add.b       #1,R6
            call        #SUBDIR             ;chama subrotina para percorrer a subArvora da a direita

SUBDIR:     mov.b       #5,R7
            mov.b       arv(R7),ARV(R6)     ;|> Imprime F
            add.b       #1,R6

            mov.b       #2,R7
            mov.b       arv(R7),ARV(R6)     ;|> Imprime C
            add.b       #1,R6

            mov.b       #6,R7
            mov.b       arv(R7),ARV(R6)     ;|> Imprime G
            add.b       #1,R6
            call        #FIM

FIM:        jmp         $


            .data
arv:        .byte       "ABCDEFGHI"

            .data
ARV:        .byte       "000000000"

;-------------------------------------------------------------------------------
; Stack Pointer definition
;-------------------------------------------------------------------------------
            .global __STACK_END
            .sect   .stack

;-------------------------------------------------------------------------------
; Interrupt Vectors
;-------------------------------------------------------------------------------
            .sect   ".reset"                ; MSP430 RESET Vector
            .short  RESET
```

Ordem Simétrica



H, D, I, B, E, A, F, C, G

```
;-------------------------------------------------------------------------------
; MSP430 Assembler Code Template for use with TI Code Composer Studio
;
;
;-------------------------------------------------------------------------------
            .cdecls C,LIST,"msp430.h"       ; Include device header file

;-------------------------------------------------------------------------------
            .def    RESET                   ; Export program entry-point to
                                            ; make it known to linker.
;-------------------------------------------------------------------------------
            .text                           ; Assemble into program memory.
            .retain                         ; Override ELF conditional linking
                                            ; and retain current section.
            .retainrefs                     ; And retain any sections that have
                                            ; references to current section.

;-------------------------------------------------------------------------------
RESET       mov.w   #__STACK_END,SP         ; Initialize stackpointer
StopWDT     mov.w   #WDTPW|WDTHOLD,&WDTCTL  ; Stop watchdog timer


;-------------------------------------------------------------------------------
;Main loop here
;Daniel Moraes da Silva - Engenharia de Computação
;Percurso Arvore: Ordem Final
;-------------------------------------------------------------------------------
OS:         mov.b       #arv,R4
            mov.b       #ARV,R5
            mov.b       #0,R6

SUBESQ:     mov.b       #7,R7               ;|
            mov.b       arv(R7),ARV(R6)     ;|> Imprime H
            add.b       #1,R6               ;|

            mov.b       #8,R7               ;|
            mov.b       arv(R7),ARV(R6)     ;|> Imprime I
            add.b       #1,R6               ;|

            mov.b       #3,R7               ;|
            mov.b       arv(R7),ARV(R6)     ;|> Imprime D
            add.b       #1,R6               ;|

            mov.b       #4,R7
            mov.b       arv(R7),ARV(R6)     ;|> Imprime E
            add.b       #1,R6

            mov.b       #1,R7
            mov.b       arv(R7),ARV(R6)     ;|> Imprime B
            add.b       #1,R6

            call        #SUBDIR             ;chama subrotina para percorrer a subArvora da a direita

SUBDIR:
            mov.b       #5,R7
            mov.b       arv(R7),ARV(R6)     ;|> Imprime F
            add.b       #1,R6

            mov.b       #6,R7
            mov.b       arv(R7),ARV(R6)     ;|> Imprime G
            add.b       #1,R6

            mov.b       #2,R7
            mov.b       arv(R7),ARV(R6)     ;|> Imprime C
            add.b       #1,R6

            mov.b       #0,R7
            mov.b       arv(R7),ARV(R6)     ;|> Imprime A
            add.b       #1,R6
            call        #FIM

FIM:        jmp         $


            .data
arv:        .byte       "ABCDEFGHI"

            .data
ARV:        .byte       "000000000"


;-------------------------------------------------------------------------------
; Stack Pointer definition
;-------------------------------------------------------------------------------
            .global __STACK_END
            .sect   .stack

;-------------------------------------------------------------------------------
; Interrupt Vectors
;-------------------------------------------------------------------------------
            .sect   ".reset"                ; MSP430 RESET Vector
            .short  RESET
```