E-fólio B | Folha de resolução para E-fólio

UNIDADE CURRICULAR: Arquitetura de Computadores

CÓDIGO: 21010

DOCENTE: Gracinda Carvalho; José Coelho; Carlos Sousa; Rui Manuel Silva

A preencher pelo estudante

NOME: Andreia Isabel Teófilo Agostinho Romão

N.º DE ESTUDANTE: 1702430

CURSO: [2105] Licenciatura em Engenharia Informática

TRABALHO / RESOLUÇÃO:

A)

Para esta alinea foi reutilizado o código da AF11 – Ex11 - Primos, algoritmo Eratosthenes. Foi testado também com o código da AF11 – Ex10 – Primos, mas decidi usar antes o do Ex11 devido à sua eficiência e consequentemente menos instruções e ciclos de relógio.

As únicas alterações feitas no código da AF:

- Tamanho da tabela auxiliar, *"dimensao EQU 9000"*, para poder alocar todos os possíveis números, de forma a conseguir ter os números primos pretendidos.

- Alterar o nome da função que termina os cálculos dos primos, ao invés de terminar o programa, este passa para a função *"RetornarA"*,

Na função *"PrimeirosPrimos"*, é garantido que o registo R1 se mantém inalterado ao sair da função e retornar ao ponto onde foi chamado.

Clock: 1 548 865

Instructions: 156 181

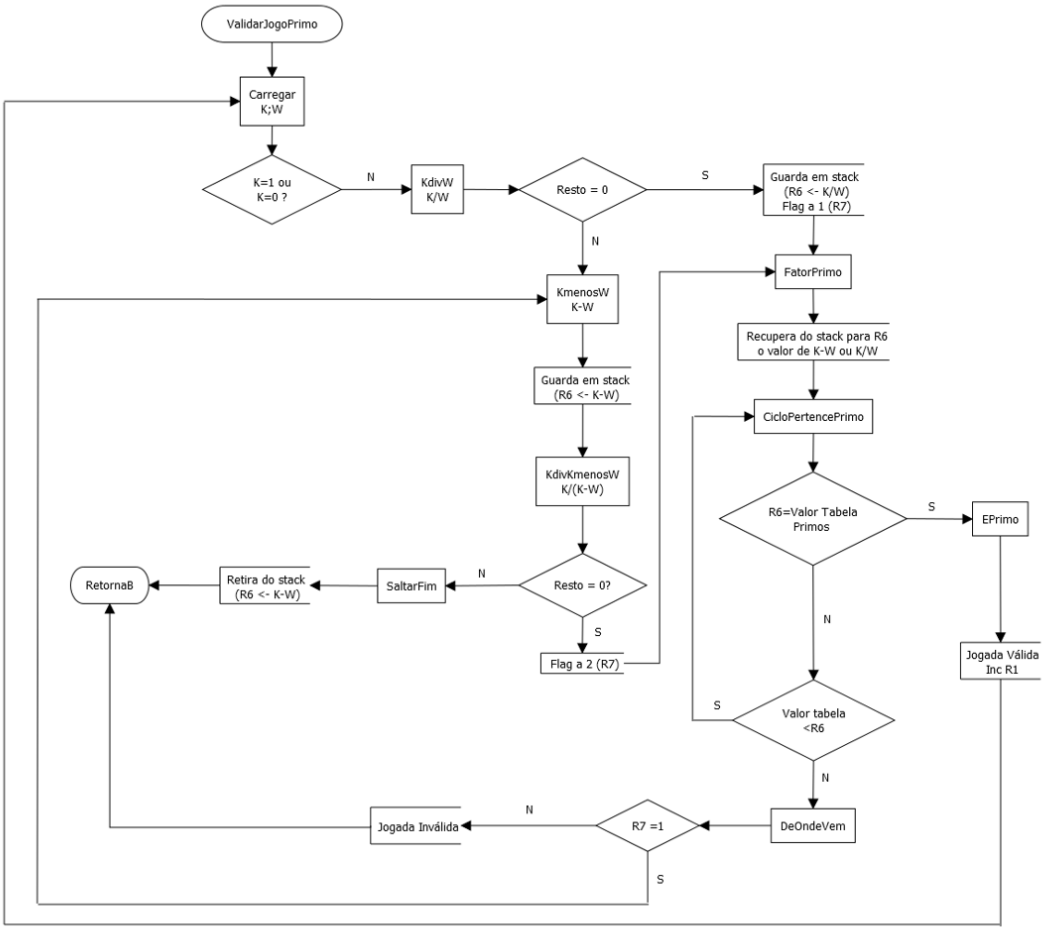
B)

Nesta alinea decidi criar um fluxograma com o funcionamento da "função" "*ValidarJogoPrimo*" uma vez que assim fica mais percetível o seu funcionamento.

Como a operação divisão altera o valor dos registos, decidi usar Registos auxiliares para K e W, para não alterar os valores lidos no início do ciclo, onde foi necessário ler da memória, poupando assim ciclos.

Uma vez que dividi a questão *em "pequenos procedimentos",* tive de garantir que tanto o resultado de K-W como K/W ficassem guardados em stack, para quando necessário estarem acessíveis.

Uma vez que teria de verificar se o resultado de K/W e se K-W seriam factores, decidi usar um registo (R7), como flag para a "função" *"FatorPrimo"* saber de qual operação pertence o possivel fator primo, ao invés de usar chamadas de "funções". Ficando assim a flag 1 para indicar que pertence a K/W e a flag 2 que pertence a K-W. Isto porque como é visível no fluxograma comecei por testar primeiro K/W e depois caso seja primo não precisa de fazer mais cálculos, mas caso o resto da divisão não seja zero ou não seja fator, fazer K/(K-W).



Clock: 1 578 104

Instructions: 158 947

C)

Para esta alinea decidi, por forma a simplificar e não ter de mexer no stack, por cada fator encontrado efetuar logo as contas pretendidas para esse primo e a cada conta efetuada contabilizar logo a jogada, guardando o resultado da soma na tabela como pretendido, retornando a procura do primo seguinte.

Na procura pelo primo, tentei otimizar o ciclo para procurar até à $\sqrt{k}$, porém não consegui, pelo que a procura é feita até o $PRIMO \geq k$.

Clock: 2 051 951

Instructions: 200 141

Anexos:

A)

Clock: 1 548 865

Instructions: 156 181

```
8000 : 1fe1 1fd3 1fbb 1fb5 1faf 1fa5 1f9d 1f99 ........
8008 : 1f97 1f91 1f85 1f7b 1f75 1f67 1f51 1f4b ........
8010 : 0000 0000 0000 0000 0000 0000 0000 0000 ........
8018 : 0000 0000 0000 0000 0000 0000 0000 0000 ........
8020 : 0000 0000 0000 0000 0000 0000 0000 0000 ........
8028 : 0000 0000 0000 0000 0000 0000 0000 0000 ........
8030 : 0000 0000 0000 0000 0000 0000 0000 0000 ........
8038 : 0000 0000 0000 0000 0000 0000 0000 0000 ........
```

B)

Clock: 1 578 104

Instructions: 158 947

```
8000 : 1fe1 1fd3 1fbb 1fb5 1faf 1fa5 1f9d 1f99 ........
8008 : 1f97 1f91 1f85 1f7b 1f75 1f67 1f51 1f4b ........
8010 : 0007 0001 0002 0002 0005 0000 0003 000b ........
8018 : 0007 0001 0006 0004 0002 0000 0003 0002 ........
8020 : 0000 0000 0000 0000 0000 0000 0000 0000 ........
8028 : 0000 0000 0000 0000 0000 0000 0000 0000 ........
8030 : 0000 0000 0000 0000 0000 0000 0000 0000 ........
8038 : 0000 0000 0000 0000 0000 0000 0000 0000 ........
```

C)

Clock: 2 051 951

Instructions: 200 141

```
8000 : 1fe1 1fd3 1fbb 1fb5 1faf 1fa5 1f9d 1f99 ........
8008 : 1f97 1f91 1f85 1f7b 1f75 1f67 1f51 1f4b ........
8010 : 0007 0001 0002 0002 0005 0000 0003 000b ........
8018 : 0007 0001 0006 0004 0002 0000 0003 0002 ........
8020 : 06c8 00f3 0001 0044 009a 003f 0065 001d ...D.?e.
8028 : 1568 502b 32e3 2169 0cca 3faa 0001 0001 ........
8030 : 0000 0000 0000 0000 0000 0000 0000 0000 ........
8038 : 0000 0000 0000 0000 0000 0000 0000 0000 ........
```

Código referente as alíneas feitas:

```
            ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;

            ; Início do bloco A vvvvvvvv não alterar para baixo vvvvvvvv

            ; zona de dados


            ORIG 8000h

resultadosA    TAB    16

resultadosB    TAB    16

resultadosC    TAB    16

resultadosD    TAB    16


            ; jogos de teste para a alínea B

Jogo1       STR    476,68,66,63,9,6,3,1,0

Jogo2       STR    476,469,156,78,39,13,3,0

Jogo3       STR    1547,119,102,83,27,1

Jogo4       STR    4389,4370,2185,87,80,40,5,1

Jogo5       STR    3990,570,30,10,2,0

Jogo6       STR    833,17,0

Jogo7       STR    798,399,392,196,49,7,1

Jogo8       STR    1254,114,112,110,108,54,27,9,6,4,2,1

Jogo9       STR    884,68,34,32,16,14,7,1

Jogo10      STR    180,60,56,28,4,2,1

Jogo11      STR    1001,143,132,66,6,3,0

Jogo12      STR    931,912,910,455,91,1

Jogo13      STR    34,17,1

Jogo14      STR    1862,1813,906,453,1

Jogo15      STR    1020,510,255,51,1

Jogo16      STR    3762,3759,537,31,1


            ; colocar jogos
```

```
Jogos       STR Jogo1,Jogo2,Jogo3,Jogo4,Jogo5,Jogo6,Jogo7,Jogo8,Jogo9,Jogo10,Jogo11,Jogo12,Jogo13,Jogo14,Jogo15,Jogo16,0


            ; números de teste para a alínea C e D

Numeros     STR 476,66,5179,34,77,24,40,12,1155,6175,2520,1716,1377,7225,7,5,0


            ; zona para retornar as jogadas

jogadas     TAB     16

            ; zona para colocar os números primos

            ; os jogos não podem ter números superiores ao maior primo

maxNumeros  EQU     1024

primos      TAB     maxNumeros


            ; Final do bloco A ^^^^^^^^^^ não alterar para cima ^^^^^^^^^^

            ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;


            ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;

; Alg AF11b-Ex11 Crivo Primos ## Alocação do espaço

            ; para os resultados e tabela

;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;


dimensao    EQU     9000

            ORIG    1000h

tabela      TAB     dimensao


            ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;

; Fim - Alg AF11b-Ex11 Crivo Primos ## Alocação do espaço

            ; para os resultados e tabela

;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
```

```
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
; Início do bloco B vvvvvvvv não alterar para baixo vvvvvvvv
; zona do código

            ORIG 0000h

; inicialização do stack
Inicio:     MOV     R1, fd1fh
            MOV     SP, R1


; TESTE alinea A
            MOV     R1, maxNumeros
            CALL    PrimeirosPrimos
            MOV     R2, resultadosA
            MOV     R4, 16

CicloA:     DEC     R1
            MOV     R5, M[R1+primos]
            MOV     M[R2], R5
            INC     R2
            DEC     R4
            BR.NZ   CicloA

FimA:       Nop

; TESTE alinea B
            MOV     R1, Jogos
```

```
            MOV     R4, R0

CicloB:     MOV     R2, M[R1]
            CMP     R2, R0
            BR.Z    FimB
            PUSH    R1
            PUSH    R4
            CALL    ValidarJogoPrimo
            POP     R4
            MOV     M[R4+resultadosB], R1
            POP     R1
            INC     R4
            INC     R1
            BR      CicloB

FimB:       Nop


            ; TESTE alinea C
            MOV     R5, Numeros
            MOV     R4, R0

CicloC:     MOV     R2, jogadas
            MOV     R1, M[R5]
            CMP     R1, R0
            BR.Z    FimC
            PUSH    R4
            PUSH    R5
            CALL    JogadasJogoPrimo
            POP     R5
            POP     R4
```

```
                MOV     M[R4+resultadosC], R0

CicloC2:        DEC     R1
                BR.N    CicloC3
                MOV     R3, M[R1+jogadas]
                ADD     M[R4+resultadosC], R3
                BR      CicloC2

CicloC3:        INC     R5
                INC     R4
                BR      CicloC

FimC:           Nop

                ; TESTE alinea D
                MOV     R5, Numeros
                MOV     R4, R0

CicloD:         MOV     R1, M[R5]
                CMP     R1, R0
                BR.Z    FimD
                PUSH    R4
                PUSH    R5
                CALL    JogoArtificialPrimo
                POP     R5
                POP     R4
                MOV     M[R4+resultadosD], R1
                INC     R5
                INC     R4
                BR      CicloD
```

```
FimD:       JMP    Fim


            ; Final do bloco B ^^^^^^^^^ não alterar para cima ^^^^^^^^^
            ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;




            ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
            ; função PrimeirosPrimos solicitada na alínea A
            ; Entrada: R1 - número de primos pretendido;
            ;        primos - endereço do local onde guardar os números
            ; Saída: resultado no endereço em R2
            ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;


            ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
; Colocar os valores dos registos que vamos alterar
; na função na pilha


PrimeirosPrimos:PUSH    R1


            ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
; Alg AF11b-Ex11 Eratosthenes ### Criação da tabela


            MOV R1, dimensao
            ADD R1, tabela
            MOV R3, tabela


SetTab:      DEC    R1
            MOV    M[R1], R3
            CMP    R1, R3
            BR.NZ  SetTab
```

```
; Fim - Alg AF11b-Ex11 Eratosthenes ### Criação da tabela
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;

        ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
; Alg AF11b-Ex11 Eratosthenes ### Preenchimento tabela primos

        MOV     R7, primos
        MOV     R5, R0
        MOV     R4, R0
        MOV     R6, dimensao
        ADD     R6, tabela
        MOV     R1, 2

NovoPrimo:   MOV     M[R7], R1
        INC     R7
        INC     R5
        CMP     R5, maxNumeros
        BR.Z    FimPrimos
        CMP     R4, R0
        BR.NZ   Prox
        MOV     R2, R1
        MOV     R3, R1
        MUL     R3, R2
        ADD     R2, tabela
        CMP     R2, R6
        BR.N    Ciclo1
        INC     R4
        BR      Prox

Ciclo1:     MOV     M[R2], R0
```

```
            ADD     R2, R1
            CMP     R2, R6
            BR.N    Ciclo1

Prox:       MOV     R2, tabela
            ADD     R2, R1
Ciclo2:     INC     R2
            INC     R1
            CMP     R2, R6
            JMP.P   FimPrimos
            CMP     M[R2],R0
            BR.Z    Ciclo2
            JMP     NovoPrimo

FimPrimos:  BR      RetornarA

; Fim - Alg AF11b-Ex11 Eratosthenes ### Preenchimento tabela primos
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;



            ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
; Repor os valores que os registos alterados na função tinham
; antes do início da mesma e
; Retorno da função para onde esta foi chamada
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;

RetornarA:  POP     R1
            RET

; Fim - PrimeirosPrimos - Alinea A
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
```

```
            ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
            ; função ValidarJogoPrimo solicitada na alínea B
            ; Entrada: R2 - endereço com as jogadas
            ;        primos - endereço com os primeiros números primos
            ; Saída: R1 - número de jogadas válidas
            ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;

ValidarJogoPrimo:MOV    R1, R0

;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
            ;Carregar valores
            ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;

Carrega:    MOV    R3, M[R2] ; R3 fica com o valor K
            INC    R2
            MOV    R4, M[R2] ; R4 fica com o valor W

;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
            ; Valida de K = 0 ou K = 1
            ; deverá de ser o primeiro a ser corrido
            ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;

            CMP    R3, R0
            JMP.Z   RetornarB
            CMP    R3, 1
            JMP.Z   RetornarB
            JMP     KdivW

;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
            ; Reg AUX
```

```
; R5 -> W

; R6 -> K

; Saida

; R6 -> Resultado da divisão

;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;

; K/W

;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;


KdivW:     MOV    R5, R4; W aux e depois será o resto da divisão

           MOV    R6, R3; K aux e depois fica com o resultado da divisão

           DIV    R6, R5; K/W

           CMP    R5, R0; Se deu resto zero testar se é primo

           BR.NZKmenosW; resto diferente de zero -> Faz K-W


           ; aqui chamar a função para validar se pertence aos primos

           PUSH    R6; Guardar R6 na pilha para depois validar se pertence aos primos

           MOV     R7, 1 ; flag

           JMP     FatorPrimo


;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;

           ; Reg AUX

           ; R5 -> W

           ; Saida

           ; R6 -> K-W

           ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;

           ; K-W

           ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;


KmenosW:   MOV    R5, R4 ; W aux

           MOV     R6, R3 ; K aux

           SUB     R6, R5 ; K-W
```

```
            PUSH    R6 ; guardar o resultado de K-W na pilha


;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;

            ; Reg AUX

            ; R5 -> K-W

            ; Saida

            ; R5 -> Resultado da divisão

            ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;

            ; K/(K-W)

            ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;


KdivKmenosW:MOV     R5, R6; K-W aux

            DIV    R3, R5; R5 vai ficar com o resultado da divisão e R3 com o resto

            CMP    R5, R0; Se deu resto zero testar se é primo

            BR.NZ   SaltarFim ;resto diferente de zero


            ; aqui chamar a função para validar se pertence aos primos

            MOV     R7, 2 ; flag

            JMP     FatorPrimo


SaltarFim:      POP     R6; já não preciso do resultado K-W, liberto-o do stack

            BR      RetornarB; resto diferente de zero - jogada invalida


;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;

            ; Reg AUX

            ; R3 -> vai interado na tabela primo

            ; R5 -> percorre a tabela dos primos

            ; R6 -> Possivel factor primo -> K-W ou K/W

            ; R7 -> marcador para saber de que operação vem se K/K-W ou K/W

            ; Saida

            ; Valida se o resultado é um factor primo
```

```
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;

; Valida se o resultado é um factor primo

;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;


FatorPrimo:     MOV     R3, primos

        POP     R6 ;recupera o valor que está na pilha pertencente aos calculos feitos em K/W ou K-W


CicloPertencePrimo:MOV     R5, M[R3]; R5 -> primo na tabela

                INC     R3

                CMP     R5, R6 ; compara se é igual

                BR.N    CicloPertencePrimo ; não é igual, repete o ciclo, o valor da tabela é menor que o
possivel factor primo

                BR.Z   EPrimo ; é primo

                BR.NN  DeOndeVem ; não é primo, o primo da tabela é maior que o possivel factor primo, logo
não vale a pena continua o teste


EPrimo:         INC     R1

                JMP     Carrega


DeOndeVem:      CMP     R7, 1

                JMP.Z   KmenosW

                JMP     RetornarB


RetornarB:      RET




                ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;

                ; função JogadasJogoPrimo solicitada na alínea C

                ; Entrada: R1 - número K a processar

                ;          R2 - endereço no qual as jogadas devem ser colocadas

                ;          primos - endereço com os primeiros números primos

                ; Saída: R1 - número de jogadas; resultado no endereço R2
```

```
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;

JogadasJogoPrimo:MOVR4, R1 ; K
MOV     R1, R0; n jogadas

            MOV     R3, primos ; fica no inicio da tabela primos
            MOV     R7, R0 ; interar até K ou >K

ProcuraFator:MOV     R5, M[R3] ; primo da tabela
            INC     R3
            CMP     R5, R4 ; compara primo com K
            BR.Z FazContas
            BR.NNRetornarC ; primo maior que K, passou procura

VerificaSeFator:MOV     R6, R4 ; K aux
            MOV    R7, R5 ; primo aux
            DIV     R6, R7 ; K/primo, R6 fica com resto, R7 resultado
            CMP     R7, R0 ;
            BR.Z    FazContas ; factor de K
            BR     ProcuraFator ; teve resto, retorna para procurar novo primo

FazContas:MOV     R6, R4 ; K aux
            MOV     R7, R5 ; primo aux

Subtrai:SUB     R6, R7 ; K-primo
            MOV     M[R2], R6 ; guarda K-primo em jogadas
            INC     R2
            INC     R1 ; contabiliza jogada

Divide:MOV     R6, R4 ; K aux
            DIV     R6, R7 ; K/primo
```

```
            MOV     M[R2], R6 ; guarda K/primo em jogadas
            INC     R2
            INC     R1 ; contabiliza jogada
            BR    ProcuraFator ; retorna para procura novo factor

RetornarC:RET



            ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
            ; função JogoArtificialPrimo solicitada na alínea D
            ; Entrada: R1 - número K a processar
            ;          primos - endereço com os primeiros números primos
            ; Saída: R1 - valor W, com a jogada para K
            ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;

JogoArtificialPrimo:    MOV     R1, R0
            RET



            ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
            ; Início do bloco C vvvvvvvv não alterar para baixo vvvvvvvv
            ; última instrução, não alterar

Fim:          JMP     Fim

            ; Final do bloco C ^^^^^^^^^ não alterar para cima ^^^^^^^^^
            ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
```