E-fólio B | Instruções para a realização do E-fólio

ARQUITETURA DE COMPUTADORES | 21010 | 2022/2023

Pretende-se implementar no P3 um programa para calcular quem ganhou o jogo dos fatores primos. O jogo dos fatores primos precisa de um número inteiro positivo K. Dois jogadores jogam à vez, de forma alternada. Em cada jogada podem dividir ou subtrair o número por um dos seus fatores primos, substituindo o valor de K e passando a vez de jogar. Ganha o jogador que fizer uma jogada convertendo o número K para 1 ou 0, ou se o adversário tiver feito uma jogada inválida. Um jogo é portanto constituído por uma sequência de números inteiros. O número 1 não é considerado primo no âmbito deste jogo.

Nas diversas alíneas pode ser reutilizado o código das atividades formativas. O código reutilizado não necessita de comentários, mas é preciso existir uma referência à atividade formativa.

a) **[1]** Pretende-se que implemente uma sub-rotina ***PrimeirosPrimos***, que coloca os primeiros N números primos em memória.
   - **Entrada:** R1 tem o número N de números primos pretendido; endereço global "primos" com o endereço no qual os números devem ser colocados.
   - **Saída**: os primeiros N números primos colocados em memória a partir do endereço "primos".
   - **Validação**: os últimos números primos irão ser copiados para a zona de validação
   - **Exemplos**: os primeiros 8 números primos são: 2h; 3h; 5h; 7h; bh; dh; 11h; 13h.

b) **[1]** Pretende-se que implemente uma sub-rotina ***ValidarJogoPrimo***, que processa um jogo com várias jogadas, e verifica quantas jogadas são válidas. Uma jogada do número K para um número W é válida se existir um movimento válido, ou seja, se K/W ou K-W é um fator primo de K.
   - **Entrada**: R2 tem o endereço E no qual as jogadas se encontram; endereço global "primos" com o endereço com os primeiros números primos, cujo último será maior ou igual ao maior número do jogo.
   - **Saída**: em R1 o número de jogadas válidas
   - **Validação**: o número de jogadas válidas irá ser copiado para a zona de validação
   - **Exemplos**: Jogo1 tem 7 jogadas válidas, parando no 1; Jogo2 tem 1 jogada válida, parando no 156 dado que não há uma jogada para este número a partir de 469. O número 469 tem apenas dois fatores primos, 7 e 67, podendo ser dividido e subtraído por um dos dois fatores primos, resultando nas seguintes jogadas: 462; 67; 402; 7.

c) **[1]** Pretende-se que implemente uma sub-rotina ***JogadasJogoPrimo***, que dado um número K calcula as jogadas possíveis.

- **Entrada**: R1 tem o número K a processar; R2 tem o endereço E no qual as jogadas devem ser colocadas; endereço global "primos" com o endereço com os primeiros números primos, cujo último será maior ou igual ao maior número do jogo.
- **Saída**: em R1 o número de jogadas; todas as jogadas colocadas a partir do endereço E. Caso um número seja válido por duas jogadas distintas, aparece duas vezes (exemplo: 6 pode ser dividido e subtraído por 2 e 3, ficando 4, 3, 3, 2)
- **Validação**: a soma de todas as jogadas possíveis (é portanto irrelevante a ordem), irá ser copiada para a zona de validação
- **Exemplos:** o número 476 tem uma soma de 6c8h, enquanto que o número 66 tem uma soma de f3h. Explicação:
    - 476 tem três fatores primos, 2, 7 e 17, ficando devido a operações de divisão e subtração com as seguintes 6 jogadas: 238 (eeh), 474 (1dah), 68 (44h), 469 (1d5h), 28 (1ch), 459 (1cbh) cuja soma é 1736 (6c8h)
    - 66 tem também três fatores primos, 2, 3 e 11, ficando com 6 jogadas: 33 (21h), 64 (40h), 22 (16h), 63 (3fh), 6 (6h), 55 (37h) cuja soma é 243 (f3h)

d) **[1]** Pretende-se que implemente uma sub-rotina ***JogoArtificialPrimo***, que dado um número K identifica uma jogada W a realizar, segundo as seguintes duas regras:

a. Se o número K for primo então W=K/K
b. De entre as jogadas possíveis para K, calcular o número de possíveis jogadas que essas jogadas permitem. Utilizar para o W o número com maior número de jogadas. Caso exista empate, W deverá ser o maior desses números

- **Entrada**: em R1 o número K; endereço global "primos" com o endereço com os primeiros números primos, cujo último será maior ou igual ao maior número do jogo.
- **Saída**: em R1 o valor W
- **Validação**: a melhor jogada irá ser copiada para a zona de validação
- **Exemplos**: o número 476 tem como melhor jogada 474 (1dah) e o número 66 tem como melhor jogada 63 (3fh). Explicação:
    - 476 tem 6 jogadas possíveis, sendo analisados o número de jogadas desses movimentos: 238 (eeh) – 6 movimentos; 474 (1dah) – 6 movimentos, 68 (44h) – 4 movimentos, 469 (1d5h) – 4 movimentos; 28 (1ch) – 4 movimentos; 459 (1cbh) – 4 movimentos. Escolhido de entre as duas possibilidades com 6 movimentos, o número mais alto
    - 66 são analisados da mesma forma os movimentos: 33 (21h) – 4 movimentos; 64 (40h) – 2 movimentos; 22 (16h) – 4 movimentos; 63 (3fh) – 4 movimentos; 6 (6h) – 4 movimentos; 55 (37h) – 4 movimentos. Escolhido de entre as cinco possibilidades de 4 movimentos, o número mais alto.

Programa de teste:

```
                ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
                ; Inicio 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 ^^^^^^^^^
                ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
```

```
                ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
                ; Inicio 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
                ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
PrimeirosPrimos: RET

                ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
                ; 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
                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:       MOV R1, R0
                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 ^^^^^^^^^
                ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
```

O programa de teste tem três blocos, A, B e C. Estes blocos não devem ser alterados, e destinam-se a operacionalizar a avaliação. Durante a avaliação podem ser utilizados outros testes, através da alteração destes blocos. Deve colocar o código das sub-rotinas, podendo alterar qualquer parte entre blocos.

Resultado de execução do programa de teste, sem o código de qualquer alínea realizada:

```
Clock: 7484
Instructions: 844
8000 : 0000 0000 0000 0000 0000 0000 0000 0000 ........
8008 : 0000 0000 0000 0000 0000 0000 0000 0000 ........
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 ........
```

BOM TRABALHO!

Avaliação

Cotação:
A cotação encontra-se junto de cada uma das alíneas, entre [].

Critérios de Correcção:
Funcionalidade: 50%
Simplicidade e Modularidade: 10%
Eficiência (serão contabilizados o número de instruções e ciclos de relógio): 10%
Apresentação do código (indentação e comentários): 10%
Relatório (Legibilidade e Justificação dos Resultados e das Opções): 20%

Descontos:
Trabalhos entregues que não estejam em conformidade com as regras de entrega do e-fólio B: até 10%
Código sem comentários, ou apenas com comentários a refletir o significado da instrução (exemplo MOV R1,R2 ;mover o conteúdo de R2 para R1) : até 50%
Deteção de fraude (total ou parcial): 100%
Trabalhos entregues após a data limite (máximo 24h) : 10%

Regras para entrega do e-fólio B:

Forma de entrega:

Deverá ser entregue um relatório em formato pdf ou Word até 5 páginas A4, com todos os cálculos e todas as opções tomadas na construção dos programas. Em anexo deve colocar todo o código e resultados obtidos (número de instruções, ciclos de relógio e conteúdo da memória de 8000h a 803fh). O código deve estar num formato que permita a seleção de modo a ser copiado e colado para o simulador do P3.

Não são aceites entregas fora da plataforma Moodle.

Ajudas extra:

Estas ajudas não fazem parte do enunciado, devem ser tomadas em conta como orientação, e utilizadas apenas se necessário.

Explicações do programa de teste:

No bloco A estão os dados de entrada. No bloco B está o código de teste das alíneas realizadas. Este código acerta os dados de entrada, e chama a rotina de cada alínea (na alínea A será uma única chamada com "maxNumeros", na alínea B será a string "Jogos", na alínea C e D será a string "Numeros". No final guarda os resultados no espaço respetivo para o efeito (a partir da posição 8000h). O bloco C destina-se a garantir que a última instrução é o final do programa, com Fim: JMP Fim.
Entre o bloco B e C, está o código das alíneas, que devem ser preenchido. Entre o bloco A e B, caso seja necessário, podem ser declaradas mais variáveis, em zonas de memória distintas das já utilizadas.

Ajuda alínea A:

Existe duas AF que geram números primos, que podem ser reutilizadas e adaptadas.

Ajuda alínea B:

Pode considerar o seguinte algoritmo:
- processar por ordem cada par de números seguidos no jogo: K e W
- sendo K 0 ou 1, terminar
- dividir e subtrair K por W
    - caso K/W não seja inteira, desconsiderar este valor
    - caso a K-W não divida K (ou seja, K/(K-W) não é inteiro), desconsiderar este valor
- verificar se os valores considerados, K/W ou K-W, algum está na lista de primos
- caso não esteja nenhum, há uma jogada inválida e termina, caso contrário segue para a próxima jogada

Ajuda alínea C:

Pode considerar o seguinte algoritmo:
- Processar os primos W de 2 a K, e para os que são divisíveis por K (K/W é inteiro):
  - Considerar W1 = K/W; W2=K-W, adicionando W1 e W2 à lista de jogadas

Ajuda alínea D:

Pode considerar o seguinte algoritmo:
- Obter a lista de jogadas, chamando a alínea C
- Para cada jogada, saber quantas jogadas permite, chamando também a alínea C, recolhendo apenas a informação do número de jogadas
- Manter a melhor jogada em cada iteração, como sendo a que tem o maior número de jogadas, ou maior valor, sendo este o valor a retornar no final