

<b>Aluno:</b>	Bruno Mariz	<b>N USP:</b>	11261826;
<b>Turma:</b>	3 (Sexta);	<b>Turno:</b>	1;
<b>Bancada:</b>	8		

dados: .word 0x1, 0x2, 3, 4, 5, 6, 7, 0x8, 0x9, 0

```

Register group: general
r0      0x21024      135204      r1      0xffffec54      -70572
r2      0xffffec5c      -70564      r3      0x103c8      66504
r4      0x10438      66616      r5      0x0      0
r6      0x102d8      66264      r7      0x0      0
r8      0x0      0      r9      0x0      0
r10     0xff7ee000      -8462336      r11     0x0      0
r12     0xffffeb78      -70792      sp      0xffffeb00      0xffffeb00

src p1/sexta-b8-t3-t1-parte1a1b-11261826.s
3      .globl main
4      main:
5          @ Inicializar parametros
B+ 6      LDR r0, =dados
>7      LDR r1, [r0, #0] @ dividendo
8
9      pronto:
10     LDR r2, =1234 @ divisor

remote Thread 1.64084 In: main      L7      PC: 0x103cc
(gdb) x/10wx $r0
0x21024:      0x00000001      0x00000002      0x00000003      0x00000004
0x21034:      0x00000005      0x00000006      0x00000007      0x00000008
0x21044:      0x00000009      0x00000000

```

```
Register group: general
r0      0x21024      135204      r1      0x1          1
r2      0xffffec5c   -70564      r3      0x103c8     66504
r4      0x10438      66616      r5      0x0         0
r6      0x102d8      66264      r7      0x0         0
r8      0x0          0          r9      0x0         0
r10     0xff7ee000   -8462336    r11     0x0         0
r12     0xffffeb78   -70792      sp      0xffffeb00     0xffffeb00

src p1/sexta-b8-t3-t1-parte1a1b-11261826.s
3      .globl main
4      main:
5      @ Inicializar parametros
B+ 6      LDR r0, =dados
7      LDR r1, [r0, #0] @ dividendo
>8      LDR r2, =1234 @ divisor
9      MOV r3, #0x0 @ quociente
10     @ Inicializar resto com dividendo

remote Thread 1.61968 In: main
(gdb) p/x $r1
```

Crie o arquivo que resolve esse problema:

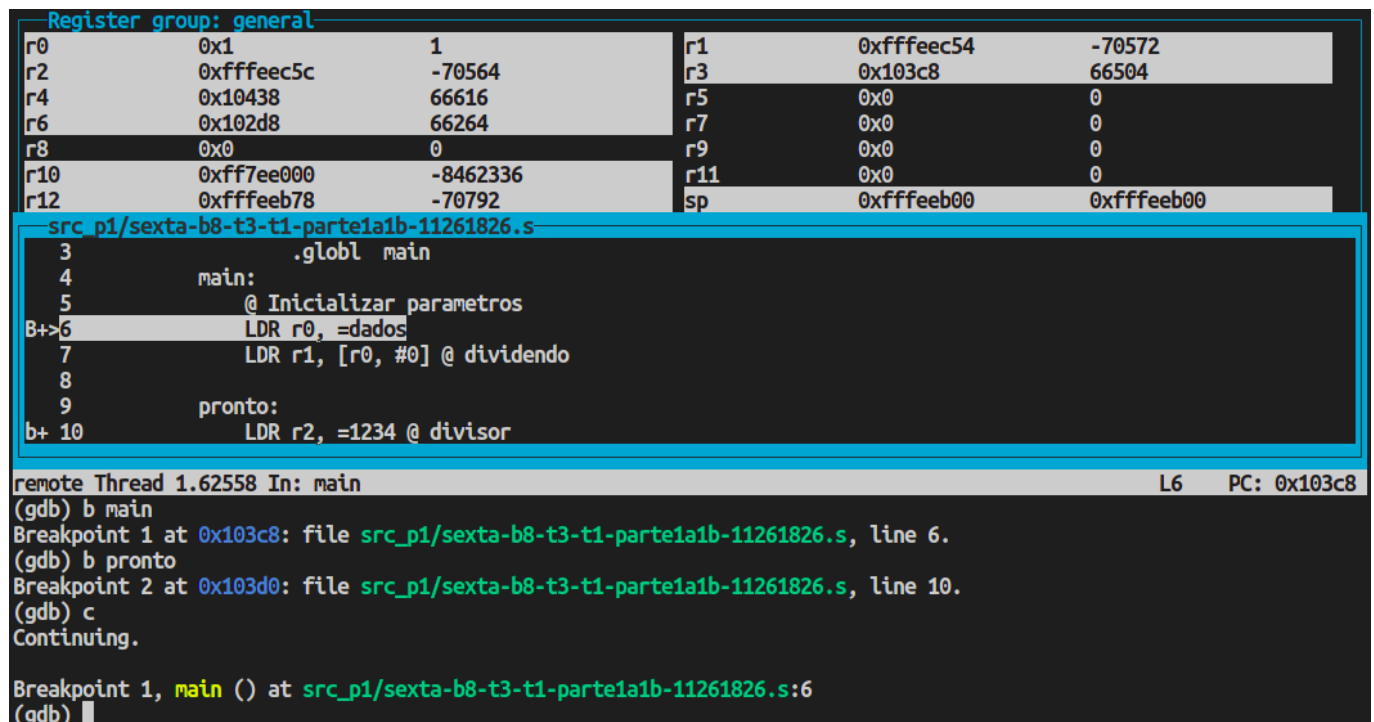
```
<dia de semana>-b<número da bancada>-t<numero da turma>-t<número do turno>-  
<partela1b>-<nusp>.s
```

Ex: sexta-b4-t3-t1-parte1a1b-9191911.s

Para rodar o ARM debug ou o gdb certifique-se de que os prints de tela sejam gerados em função de comandos apropriados, labels, resultados, como segue:

# Geralmente paramos em main (break main). Dessa vez vamos parar em pronto também. Entre main e pronto inicialize R0 com o endereço de “dados” e R1 com o primeiro elemento de dados.

```
b main  
b pronto  
c
```



The screenshot shows the GDB interface with the following content:

**Register group: general**

Register	Value	Comment
r0	0x1	1
r1	0xffffec54	-70572
r2	0xffffec5c	-70564
r3	0x103c8	66504
r4	0x10438	66616
r5	0x0	0
r6	0x102d8	66264
r7	0x0	0
r8	0x0	0
r9	0x0	0
r10	0xff7ee000	-8462336
r11	0x0	0
r12	0xfffeeb78	-70792
sp	0xfffeeb00	0xfffeeb00

**src\_p1/sexta-b8-t3-t1-parte1a1b-11261826.s**

```
3      .globl main  
4      main:  
5          @ Inicializar parametros  
B->6      LDR r0, =dados  
7          LDR r1, [r0, #0] @ dividendo  
8  
9      pronto:  
b+ 10      LDR r2, =1234 @ divisor
```

**remote Thread 1.62558 In: main** L6 PC: 0x103c8

```
(gdb) b main  
Breakpoint 1 at 0x103c8: file src_p1/sexta-b8-t3-t1-parte1a1b-11261826.s, line 6.  
(gdb) b pronto  
Breakpoint 2 at 0x103d0: file src_p1/sexta-b8-t3-t1-parte1a1b-11261826.s, line 10.  
(gdb) c  
Continuing.  
  
Breakpoint 1, main () at src_p1/sexta-b8-t3-t1-parte1a1b-11261826.s:6  
(gdb)
```

C

```

Register group: general
r0      0x21024      135204      r1      0x1          1
r2      0xfffec5c    -70564      r3      0x103c8     66504
r4      0x10438      66616      r5      0x0          0
r6      0x102d8      66264      r7      0x0          0
r8      0x0           0          r9      0x0          0
r10     0xff7ee000   -8462336    r11     0x0          0
r12     0xfffeb78    -70792      sp      0xfffeb00     0xfffeb00

src_p1/sexta-b8-t3-t1-parte1a1b-11261826.s
7      LDR r1, [r0, #0] @ dividendo
8
9      pronto:
B->10   LDR r2, =1234 @ divisor
11     MOV r3, #0x0 @ quociente
12     @ Inicializar resto com dividendo
13     MOV r5, r1 @ resto
14     @ Salva divisor original

remote Thread 1.62558 In: pronto L10 PC: 0x103d0
(gdb) c
Continuing.

Breakpoint 1, main () at src_p1/sexta-b8-t3-t1-parte1a1b-11261826.s:6
(gdb) c
Continuing.

Breakpoint 2, pronto () at src_p1/sexta-b8-t3-t1-parte1a1b-11261826.s:10
(gdb)

```

# Imprima o valor de R0 para ver o endereço de dados.

```
p/x $r0
```

```

Register group: general
r0      0x21024      135204      r1      0x1          1
r2      0xfffec5c    -70564      r3      0x103c8     66504
r4      0x10438      66616      r5      0x0          0
r6      0x102d8      66264      r7      0x0          0
r8      0x0           0          r9      0x0          0
r10     0xff7ee000   -8462336    r11     0x0          0
r12     0xfffeb78    -70792      sp      0xfffeb00     0xfffeb00

src_p1/sexta-b8-t3-t1-parte1a1b-11261826.s
7      LDR r1, [r0, #0] @ dividendo
8
9      pronto:
B->10   LDR r2, =1234 @ divisor
11     MOV r3, #0x0 @ quociente
12     @ Inicializar resto com dividendo
13     MOV r5, r1 @ resto
14     @ Salva divisor original

remote Thread 1.62558 In: pronto L10 PC: 0x103d0
(gdb) p/x $r0
$1 = 0x21024

```

# Imprima o valor de R1 para ver o valor do primeiro elemento de dados.

```
p/x $r1
```

```

Register group: general
r0      0x21024      135204      r1      0x1          1
r2      0xffffec5c   -70564      r3      0x103c8     66504
r4      0x10438      66616      r5      0x0         0
r6      0x102d8      66264      r7      0x0         0
r8      0x0          0          r9      0x0         0
r10     0xff7ee000   -8462336    r11     0x0         0
r12     0xffffeb78   -70792      sp      0xffffeb00   0xffffeb00

src p1/sexta-b8-t3-t1-parte1a1b-11261826.s
7      LDR r1, [r0, #0] @ dividendo
8
9      pronto:
B->10   LDR r2, =1234 @ divisor
11     MOV r3, #0x0 @ quociente
12     @ Inicializar resto com dividendo
13     MOV r5, r1 @ resto
14     @ Salva divisor original

remote Thread 1.62558 In: pronto L10 PC: 0x103d0
(gdb) p/x $r1
$8 = 0x1

```

Use comando de leitura de memória para checar se no endereço de dados R0 está o valor de dados igual de R1. Gere prints de tela com os resultados.

```

Register group: general
r0      0x21024      135204      r1      0x1          1
r2      0xffffec5c   -70564      r3      0x103c8     66504
r4      0x10438      66616      r5      0x0         0
r6      0x102d8      66264      r7      0x0         0
r8      0x0          0          r9      0x0         0
r10     0xff7ee000   -8462336    r11     0x0         0
r12     0xffffeb78   -70792      sp      0xffffeb00   0xffffeb00

src p1/sexta-b8-t3-t1-parte1a1b-11261826.s
7      LDR r1, [r0, #0] @ dividendo
8
9      pronto:
B->10   LDR r2, =1234 @ divisor
11     MOV r3, #0x0 @ quociente
12     @ Inicializar resto com dividendo
13     MOV r5, r1 @ resto
14     @ Salva divisor original

remote Thread 1.63736 In: pronto L10 PC: 0x103d0
(gdb) x/x $r0
0x21024: 0x00000001
(gdb) x/10wx $r0
0x21024: 0x00000001 0x00000002 0x00000003 0x00000004
0x21034: 0x00000005 0x00000006 0x00000007 0x00000008
0x21044: 0x00000009 0x00000000

```

2. (6.0) Altere o código da parte 1. Resolva a parte 2 com R0 = 3. Suponha apenas a título de exemplo que, R0 contém o valor 2 (R0 = 2). Varra todos os elementos de dados e sinalize aqueles para os quais 2 é um divisor inteiro, ou seja, o valor do elemento de dados dividido por 2 resulta m inteiro, com resto 0, sinalizando com 1. Sinalize com 0 em caso contrário. Gere uma nova sequência de words declarada como:

resposta: .space 88 @abre espaço de 88 bytes para armazenar a resposta

Exemplo: Para R0 = 2, em resposta devemos ter: 0,1,0,1,0,1,0,1,0,-1

Onde -1 indica o término da sequencia em "resposta". A interpretação é que 2 é divisor inteiro de 2, 4, 6, 8, e não é divisor inteiro de 1, 3, 5, 7, 9.

Crie o arquivo que resolve esse problema:

```

<dia de semana>-b<número da bancada>-t<numero da turma>-t<número do turno>-
<parte2>-<nusp>.s

```

Ex: sexta-b4-t3-t1-parte2-9191911.s

Gere prints de tela com os resultados.