Ano Lectivo 2013/14 - 1º Semestre

Mini-Teste Prático 1 (exemplo de resolução)

NOTE BEM: Leia atentamente todas as questões, comente o código usando a linguagem C e respeite a convenção de passagem de parâmetros e salvaguarda de registos que estudou. Na tradução para o *Assembly* do MIPS respeite rigorosamente os aspectos estruturais e a sequência de instruções indicadas no código original fornecido.
O código em C apresentado pode não estar funcionalmente correcto, pelo que **não deve ser interpretado**.

1) Codifique em *assembly* do MIPS a seguinte função `main`:

```
void main(void)
{
  int i, j;        // i => $t0; j => $t1
  int k = 0;       // k => $t2

  print_string("Dois números: ");
                                // syscall
  i = read_int();               // syscall
  j = read_int();               // syscall
  if (i != 0)
    i = (i << 1) & 0xF;
  do
  {
    if ((i < j) && (k > j))
      j = j & k;
    else
      j = j ^ k;
    k++;
  } while(k < 5);
}
```

Label	Instrução em *assembly*	Comentário em C
	`.data`	
`str:`	`.asciiz "Dois Números: "`	
	`.text`	
	`.globl main`	
`main:`		`# void main (void)`
		`# {`
		`#  int i, j;`
	`li  $t2, 0`	`#  int k = 0;`
	`li  $v0, 4`	
	`la  $a0, str`	
	`syscall`	`#  print_string("Dois números: ");`
	`li  $v0, 5`	
	`syscall`	
	`move  $t0, $v0`	`#  i = read_int();`
	`li  $v0, 5`	
	`syscall`	
	`move  $t1, $v0`	`#  j = read_int();`

Label	Instrução em *assembly*	Comentário em C
`if1:`	`beqz  $t0, do`	`# if (i != 0)`
	`sll  $t0, $t0, 1`	
	`andi  $t0, $t0, 0xF`	`#  i = (i << 1) & 0xF;`
`do:`		`# do`
		`# {`
`if2:`	`bge  $t0, $t1, else2`	`#  if ((i < j) && (k > j))`
	`ble  $t2, $t1, else2`	
	`and  $t1, $t1, $t2`	`#  j = j & k;`
	`j  endif2`	
`else2:`		`#  else`
	`xor  $t1, $t1, $t2`	`#  j = j ^ k;`
`endif2:`		
	`addi  $t2, $t2, 1`	`#  k++`
	`blt  $t2, 5, do`	`# } while(k < 5);`
	`jr  $ra`	`# }`

Nº Mec.: _____ Nome: _____ Turma: _____

2) Codifique em *assembly* do MIPS a seguinte função `main`:

```
void main(void)
{
  static int a[11], b[11];
  int i, *p, *m;
  for(i = 0; i <= 10; i++)
    a[i] = i - 5;
  p = a;
  m = b + 10;
  while(p < m)
  {
    if(*p < 0)
      *m = 5 - (*p);
    else
      *m = *(p + 1);
    p++;
    m--;
  }
  print_int10(b[8]);    // syscall
}
```

Label	Instrução em *assembly*	Comentário em C
	.data	
	.align 2	# opcional
a:	.space 44	# 11 * 4
	.align 2	# opcional
b:	.space 44	
	.text	# i em $t0
	.globl main	# p em $t1
		# m em $t2
main:		# void main (void)
		# {
for0:	li $t0, 0	# for(i=0;i<=10;i++)
test0:	bgt $t0, 10, end0	
body0:	subi $t9, $t0, 5	
	la $t8, a	
	sll $t7, $t0, 2	
	addu $t8, $t8, $t7	
	sw $t9, 0($t8)	# a[i] = i - 5;
next0:	addi $t0, $t0, 1	
	j test0	
end0:		
	la $t1, a	# p = a;
	la $t2, b + 40	# m = b + 10;

Label	Instrução em *assembly*	Comentário em C
while1:	bgeu $t1, $t2, end1	# while(p < m)
body1:		# {
if2:	lw $t9, 0($t1)	
	bge $t9, 0, else2	# if(*p < 0)
then2:	li $t8, 5	
	sub $t9, $t8, $t9	
	sw $t9, 0($t2)	# *m = 5 - (*p);
	j end2	
else2:		# else
	lw $t9, 4($t1)	
	sw $t9, 0($t2)	# *m = *(p+1);
end2:		
	addiu $t1, $t1, 4	# p++;
	subiu $t2, $t2, 4	# m--;
	j while1	
end1:		
	la $t9, b	
	lw $a0, 32($t9)	
	li $v0, 1	
	syscall	# print_int10(b[8]);
	jr $ra	}