ECE/CS 314
Spring 2004

Section 3

1st Attempt at Supporting Functions

Memory

0x7FFFFFFC				
0x7FFFFFF8				
0x7FFFFFF4				
0x7FFFFFF0				
0x7FFFFFEC				
0x7FFFFFE8				
0x7FFFFFE4				
0x7FFFFFE0				
0x7FFFFFDC				
0x7FFFFFF8				
• • •		• • •		
0x10000010				
0x1000000C				
0x10000008	‘U’	‘N’	\0	
0x10000004	‘I’	‘S’	‘ ’	‘F’
0x10000000	‘3’	‘1’	‘4’	‘ ’

Registers

0	0x00000000	16	
1		17	
2		18	
3		19	
4	0x10000000	20	
5		21	
6		22	
7		23	
8		24	
9		25	
10		26	
11		27	
12		28	
13		29	
14		30	
15		31	

```
Caller:     li      $4,0x10000000
            j       NumSpaces
Return:             ...

...

NumSpaces:  addu    $17,$4,$0       #get passed argument
            li      $16,0           #count=0
$start:     lbu     $8,0($17)       #temp = *s
            beq     $8,$0,$done     #if *s==0 go to done
            li      $9,32           #temp2 = ‘ ‘
            bne     $8,$9,$skipinc  #if *s != ‘ ‘ go to skipinc
            addiu   $16,$16,1       #count++
$skipinc:   addiu   $17,$17,1       #s++
            beq     $0,$0,$start    #go to start
$done:      addu    $2,$0,$16       #Return count
            j       Return          #Return from function
```

2st Attempt at Supporting Functions

Memory

0x7FFFFFFC				
0x7FFFFFF8				
0x7FFFFFF4				
0x7FFFFFF0				
0x7FFFFFEC				
0x7FFFFFE8				
0x7FFFFFE4				
0x7FFFFFE0				
0x7FFFFFDC				
0x7FFFFFF8				
•••		•••		
0x10000010				
0x1000000C				
0x10000008	‘U’	‘N’	\0	
0x10000004	‘I’	‘S’	‘ ’	‘F’
0x10000000	‘3’	‘1’	‘4’	‘ ’

Registers

0	0x00000000	16	
1		17	
2		18	
3		19	
4	0x10000000	20	
5		21	
6		22	
7		23	
8		24	
9		25	
10		26	
11		27	
12		28	
13		29	
14		30	
15		31	0x00000044

```
Caller:     li      $4,0x10000000  #e.g. PC = 0x38
            jal     NumSpaces      #call NumSpaces (PC = 0x3C)

...

NumSpaces:  addu    $17,$4,$0      #get passed argument
            li      $16,0          #count=0
$start:     lbu     $8,0($17)      #temp = *s
            beq     $8,$0,$done    #if *s==0 go to done
            li      $9,32          #temp2 = ‘ ‘
            bne     $8,$9,$skipinc #if *s != ‘ ‘ go to skipinc
            addiu   $16,$16,1      #count++
$skipinc:   addiu   $17,$17,1      #s++
            beq     $0,$0,$start   #go to start
$done:      addu    $2,$0,$16      #Return count
            jr      $31            #Return from function
```

What About Recursion? Lets Trace It!

Memory

0x7FFFFFFC				
0x7FFFFFF8				
0x7FFFFFF4				
0x7FFFFFF0				
0x7FFFFFEC				
0x7FFFFFE8				
0x7FFFFFE4				
0x7FFFFFE0				
0x7FFFFFDC				
0x7FFFFFF8				
• • •		• • •		
0x10000010				
0x1000000C				
0x10000008	‘U’	‘N’	\0	
0x10000004	‘I’	‘S’	‘ ’	‘F’
0x10000000	‘3’	‘1’	‘4’	‘ ’

Registers

0	0x00000000	16	
1		17	
2		18	
3		19	
4	0x10000000	20	
5		21	
6		22	
7		23	
8		24	
9		25	
10		26	
11		27	
12		28	
13		29	
14		30	
15		31	0x00000044

```
Caller:     li      $4,0x10000000  #e.g. PC = 0x38
            jal     NumSpaces      #call NumSpaces (PC = 0x3C)

...

NumSpaces:  addu    $17,$4,$0      #s = argument1
            lbu     $8,0($17)      #temp = *s
            beq     $8,$0,$done    #if *s == 0 go to done
            addiu   $4,$4,1        #argument1 = s + 1
            jal     NumSpaces      #call NumSpaces

            li      $9,32          #temp2 = ‘ ‘
            bne     $8,$9,$skipinc #if *s != ‘ ‘ go to skipinc

            addiu   $2,$2,1        #count++
$skipinc:   jr      $31            #return
$done:      li      $2,0           #return value = 0
            jr      $31            #return
```

Memory

0x7FFFFFFC				
0x7FFFFFF8				
0x7FFFFFF4				
0x7FFFFFF0				
0x7FFFFFEC				
0x7FFFFFE8				
0x7FFFFFE4				
0x7FFFFFE0				
0x7FFFFFDC				
0x7FFFFFF8				
• • •		• • •		
0x10000010				
0x1000000C				
0x10000008	‘U’	‘N’	\0	
0x10000004	‘I’	‘S’	‘ ’	‘F’
0x10000000	**‘3’**	‘1’	‘4’	‘ ’

Registers

0	0x00000000	16	
1		17	**0x10000000**
2		18	
3		19	
4	**0x10000001**	20	
5		21	
6		22	
7		23	
8	**‘3’**	24	
9		25	
10		26	
11		27	
12		28	
13		29	
14		30	
15		31	0x00000044

```
Caller:     li      $4,0x10000000  #e.g. PC = 0x38
            jal     NumSpaces      #call NumSpaces (PC = 0x3C)

...

NumSpaces:  addu    $17,$4,$0      #s = arg1 (e.g. PC=0xD0)
            lbu     $8,0($17)      #temp = *s
            beq     $8,$0,$done    #if *s == 0 go to done
            addiu   $4,$4,1        #arg1 = s + 1
            jal     NumSpaces      #call NumSpaces

            li      $9,32          #temp2 = ‘ ‘
            bne     $8,$9,$skipinc #if *s != ‘ ‘ go to skipinc

            addiu   $2,$2,1        #count++
$skipinc:   jr      $31            #return
$done:      li      $2,0           #return value = 0
            jr      $31            #return
```

Memory

0x80000000				
0x7FFFFFFC				
0x7FFFFFF8				
0x7FFFFFF4				
0x7FFFFFF0				
0x7FFFFFEC				
0x7FFFFFE8				
0x7FFFFFE4				
0x7FFFFFE0				
0x7FFFFFDC				
• • •		• • •		
0x10000010				
0x1000000C				
0x10000008	‘U’	‘N’	\0	
0x10000004	‘I’	‘S’	‘ ’	‘F’
0x10000000	**‘3’**	**‘1’**	‘4’	‘ ’

Registers

0	0x00000000	16	
1		17	0x10000000
2		18	
3		19	
4	0x10000001	20	
5		21	
6		22	
7		23	
8	‘3’	24	
9		25	
10		26	
11		27	
12		28	
13		29	
14		30	
15		31	**0x000000E8**


```
NumSpaces:  addiu   $29,$29,-4
            sw      $31,0($29)      #save return address
            addu    $17,$4,$0       #s = arg1 (e.g. PC=0xD0)
            lbu     $8,0($17)       #temp = *s
            beq     $8,$0,$done     #if *s == 0 go to done
            addiu   $4,$4,1         #arg1 = s + 1
            jal     NumSpaces       #call NumSpaces (PC=0xE0)

            li      $9,32           #temp2 = ‘ ‘ (PC=0xE8)
            bne     $8,$9,$skipinc  #if *s != ‘ ‘ go to skipinc

            addiu   $2,$2,1         #count++
$skipinc:   jr      $31             #return
$done:      li      $2,0            #return value = 0
            jr      $31             #return
```

Memory

0x80000000				
0x7FFFFFFC				
0x7FFFFFF8				
0x7FFFFFF4				
0x7FFFFFF0				
0x7FFFFFEC				
0x7FFFFFE8				
0x7FFFFFE4				
0x7FFFFFE0				
0x7FFFFFDC				
• • •		• • •		
0x10000010				
0x1000000C				
0x10000008	‘U’	‘N’	\0	
0x10000004	‘I’	‘S’	‘ ’	‘F’
0x10000000	‘3’	‘1’	‘4’	‘ ’

Registers

0	0x00000000	16	
1		17	
2		18	
3		19	
4	0x10000000	20	
5		21	
6		22	
7		23	
8		24	
9		25	
10		26	
11		27	
12		28	
13		29	**0x80000000**
14		30	
15		31	0x00000044

```
NumSpaces:  addiu   $29,$29,-4      #allocate space (1 word)
            sw      $31,0($29)      #save return address
            addu    $17,$4,$0       #s = arg1 (e.g. PC=0xD0)
            lbu     $8,0($17)       #temp = *s
            beq     $8,$0,$done     #if *s == 0 go to done
            addiu   $4,$4,1         #arg1 = s + 1
            jal     NumSpaces       #recursive call(PC=0xE0)
            li      $9,32           #temp2 = ‘ ‘ (PC=0xE8)
            bne     $8,$9,$skipinc  #if *s != ‘ ‘ go to skipinc
            addiu   $2,$2,1         #count++
$skipinc:   lw      $31,0($29)      #pop return address
            addiu   $29,$29,4       #deallocate stack space
            jr      $31             #return
$done:      li      $2,0            #return value = 0
            lw      $31,0($29)      #pop return address
            addiu   $29,$29,4       #deallocate stack space
            jr      $31             #return
```

Memory

0x80000000				
0x7FFFFFFC	**00**	**00**	**00**	**44**
0x7FFFFFF8				
0x7FFFFFF4				
0x7FFFFFF0				
0x7FFFFFEC				
0x7FFFFFE8				
0x7FFFFFE4				
0x7FFFFFE0				
0x7FFFFFDC				
• • •		• • •		
0x10000010				
0x1000000C				
0x10000008	‘U’	‘N’	\0	
0x10000004	‘I’	‘S’	‘ ’	‘F’
0x10000000	**‘3’**	‘1’	‘4’	‘ ’

Registers

0	0x00000000	16	
1		17	0x10000000
2		18	
3		19	
4	0x10000001	20	
5		21	
6		22	
7		23	
8	‘3’	24	
9		25	
10		26	
11		27	
12		28	
13		29	**0x7FFFFFFC**
14		30	
15		31	0x00000044

```
NumSpaces:  addiu   $29,$29,-4      #allocate space (1 word)
            sw      $31,0($29)      #save return address
            addu    $17,$4,$0       #s = arg1 (e.g. PC=0xD0)
            lbu     $8,0($17)       #temp = *s
            beq     $8,$0,$done     #if *s == 0 go to done
            addiu   $4,$4,1         #arg1 = s + 1
            jal     NumSpaces       #recursive call(PC=0xE0)
            li      $9,32           #temp2 = ‘ ‘ (PC=0xE8)
            bne     $8,$9,$skipinc  #if *s != ‘ ‘ go to skipinc
            addiu   $2,$2,1         #count++
$skipinc:   lw      $31,0($29)      #pop return address
            addiu   $29,$29,4       #deallocate stack space
            jr      $31             #return
$done:      li      $2,0            #return value = 0
            lw      $31,0($29)      #pop return address
            addiu   $29,$29,4       #deallocate stack space
            jr      $31             #return
```

Memory

0x80000000				
0x7FFFFFFC	00	00	00	44
0x7FFFFFF8	**00**	**00**	**00**	**E8**
0x7FFFFFF4				
0x7FFFFFF0				
0x7FFFFFEC				
0x7FFFFFE8				
0x7FFFFFE4				
0x7FFFFFE0				
0x7FFFFFDC				
• • •		• • •		
0x10000010				
0x1000000C				
0x10000008	‘U’	‘N’	\0	
0x10000004	‘I’	‘S’	‘ ’	‘F’
0x10000000	**‘3’**	**‘1’**	‘4’	‘ ’

Registers

0	0x00000000	16	
1		17	**0x10000001**
2		18	
3		19	
4	**0x10000002**	20	
5		21	
6		22	
7		23	
8	**‘1’**	24	
9		25	
10		26	
11		27	
12		28	
13		29	**0x7FFFFFF8**
14		30	
15		31	**0x000000E8**

```
NumSpaces:  addiu   $29,$29,-4      #allocate space (1 word)
            sw      $31,0($29)      #save return address
            addu    $17,$4,$0       #s = arg1 (e.g. PC=0xD0)
            lbu     $8,0($17)       #temp = *s
            beq     $8,$0,$done     #if *s == 0 go to done
            addiu   $4,$4,1         #arg1 = s + 1
            jal     NumSpaces       #recursive call(PC=0xE0)
            li      $9,32           #temp2 = ‘ ‘ (PC=0xE8)
            bne     $8,$9,$skipinc  #if *s != ‘ ‘ go to skipinc
            addiu   $2,$2,1         #count++
$skipinc:   lw      $31,0($29)      #pop return address
            addiu   $29,$29,4       #deallocate stack space
            jr      $31             #return
$done:      li      $2,0            #return value = 0
            lw      $31,0($29)      #pop return address
            addiu   $29,$29,4       #deallocate stack space
            jr      $31             #return
```

Memory

0x80000000				
0x7FFFFFFC	00	00	00	44
0x7FFFFFF8	00	00	00	E8
0x7FFFFFF4	00	00	00	E8
0x7FFFFFF0	00	00	00	E8
0x7FFFFFEC	00	00	00	E8
0x7FFFFFE8	00	00	00	E8
0x7FFFFFE4	00	00	00	E8
0x7FFFFFE0	00	00	00	E8
0x7FFFFFDC	00	00	00	E8
• • •		• • •		
0x10000010				
0x1000000C				
0x10000008	‘U’	‘N’	\0	
0x10000004	‘I’	‘S’	‘ ’	‘F’
0x10000000	‘3’	‘1’	‘4’	‘ ’

Registers

0	0x00000000	16	
1		17	0x1000000A
2	0x00000000	18	
3		19	
4	0x1000000A	20	
5		21	
6		22	
7		23	
8	0x00000000	24	
9		25	
10		26	
11		27	
12		28	
13		29	0x7FFFFFD8
14		30	
15		31	0x000000E8


```
NumSpaces:  addiu   $29,$29,-4      #allocate space (1 word)
            sw      $31,0($29)      #save return address
            addu    $17,$4,$0       #s = arg1 (e.g. PC=0xD0)
            lbu     $8,0($17)       #temp = *s
            beq     $8,$0,$done     #if *s == 0 go to done
            addiu   $4,$4,1         #arg1 = s + 1
            jal     NumSpaces       #recursive call(PC=0xE0)
            li      $9,32           #temp2 = ‘ ‘ (PC=0xE8)
            bne     $8,$9,$skipinc  #if *s != ‘ ‘ go to skipinc
            addiu   $2,$2,1         #count++
$skipinc:   lw      $31,0($29)      #pop return address
            addiu   $29,$29,4       #deallocate stack space
            jr      $31             #return
$done:      li      $2,0            #return value = 0
            lw      $31,0($29)      #pop return address
            addiu   $29,$29,4       #deallocate stack space
            jr      $31             #return
```

Memory

0x80000000				
0x7FFFFFFC				
0x7FFFFFF8	00	00	00	44
0x7FFFFFF4	'3'			
0x7FFFFFF0	00	00	00	E8
0x7FFFFFEC	'1'			
0x7FFFFFE8	00	00	00	E8
0x7FFFFFE4	'4'			
0x7FFFFFE0	00	00	00	E8
0x7FFFFFDC	' '			
• • •		• • •		
0x10000010				
0x1000000C				
0x10000008	'U'	'N'	\0	
0x10000004	'I'	'S'	' '	'F'
0x10000000	'3'	'1'	'4'	' '

Registers

0	0x00000000	16	
1		17	0x1000000A
2	0x00000000	18	
3		19	
4	0x1000000A	20	
5		21	
6		22	
7		23	
8	0x00000000	24	
9		25	
10		26	
11		27	
12		28	
13		29	0x7FFFFFD8
14		30	
15		31	0x000000E8

```
NumSpaces:  addiu   $29,$29,-8      #allocate space (2 word)
            sw      $8,-4($29)      #save $8...
            sw      $31,0 ($29)     #save return address
            addu    $17,$4,$0       #s = arg1 (e.g. PC=0xD0)
            lbu     $8,0($17)       #temp = *s
            beq     $8,$0,$done     #if *s == 0 go to done
            addiu   $4,$4,1         #arg1 = s + 1
            jal     NumSpaces       #recursive call(PC=0xE0)
            li      $9,32           #temp2 = ' ' (PC=0xE8)
            bne     $8,$9,$skipinc  #if *s != ' ' go to skipinc
            addiu   $2,$2,1         #count++
$skipinc:   lw      $8,-4($29)      #pop $8
            lw      $31,0($29)      #pop return address
            addiu   $29,$29,8       #deallocate stack space
            jr      $31             #return
$done:      li      $2,0            #return value = 0
            lw      $8,-4($29)      #pop $8
            lw      $31,0($29)      #pop return address
```