|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Service** | **Code in $v0** | E-mail

|  |
| --- |
| **Arguments** |

 | **Result** |
| print integer | 1 | $a0 = integer to print |   |
| print float | 2 | $f12 = float to print |   |
| print double | 3 | E-mail

|  |
| --- |
| $f12 = double to print |

 |   |
| print string | 4 | $a0 = address of null-terminated string to print |   |
| read integer | 5 |   | $v0 contains integer read |
| read float | 6 | E-mail | $f0 contains float read |
| read double | 7 |   | $f0 contains double read |
| read string | 8 | $a0 = address of input buffer | *See note below table* |
| $a1 = maximum number of characters to read |
| sbrk (allocate heap memory) | 9 | $a0 = number of bytes to allocate | $v0 contains address of allocated memory |
| exit (terminate execution) | 10 |   |   |
| print character | 11 | $a0 = character to print | *See note below table* |
| read character | 12 |   | $v0 contains character read |

**How to use SYSCALL system services**

Step 1. Load the service number in register $v0.
Step 2. Load argument values, if any, in $a0, $a1, $a2, or $f12 as specified.
Step 3. Issue the SYSCALL instruction.
Step 4. Retrieve return values, if any, from result registers as specified.

**Example: display the value stored in $t0 on the console**

 li $v0, 1 # service 1 is print integer

.data

 outText: .asciiz "MIPS for C=A+B.\n Total stored in C is: "

 A: .word 0

 B: .word 0

 C: .word 0

 prompt1: .asciiz "Enter the number to store in A: "

 prompt2: .asciiz "\Enter the number to store in B: "

.text

 li $v0,4 #get ready to display the message

 la $a0,prompt1

 syscall #displays the first prompt

 #read the first number and save it in A

 li $v0,5 #5 to read integer

 syscall

 #move $t1,$v0

 sw $v0, A

 lw $t1,A

 li $v0,4 #get ready to display the message

 la $a0,prompt2

 syscall #displays the second prompt

 #read the second number and save it in B

 li $v0,5 #5 to read integer

 syscall

 sw $v0, B

 lw $t2, B

 add $t3,$t2,$t1

 sw $t3, C

 li $v0,4

 la $a0, outText

 syscall

 li $v0,1

 move $a0,$t3

 syscall