To request service, SPIM simulator asks the program to load the system call (syscall) code into register $v0, and the arguments into registers $a0-$a3.

Table 1: System Services

<table>
<thead>
<tr>
<th>Service</th>
<th>Code</th>
<th>Argument</th>
<th>Result</th>
</tr>
</thead>
<tbody>
<tr>
<td>print_int</td>
<td>1</td>
<td>$a0 = integer</td>
<td></td>
</tr>
<tr>
<td>print_string</td>
<td>4</td>
<td>$a0 = string</td>
<td></td>
</tr>
<tr>
<td>read_int</td>
<td>5</td>
<td></td>
<td>Integer in $a0</td>
</tr>
<tr>
<td>exit</td>
<td>10</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

```
.data
str: .asciiz "the answer = "

.text
li $v0, 4       # system call code for print_str
la $a0, str     # address of string to print
syscall        # print the string

li $v0, 1       # system call code for print_int
li $a0, 5       # integer to print
syscall        # print the integer
```
MIPS Assembler Directives

SPIM supports a subset of the MIPS assembler directives. Some important ones are:

<table>
<thead>
<tr>
<th>Directive</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>.data &lt;addr&gt;</td>
<td># store subsequent items in data segment, starting at optional address.</td>
</tr>
<tr>
<td>.text &lt;addr&gt;</td>
<td># store subsequent items in text segment, starting at optional address.</td>
</tr>
<tr>
<td>.asciiz str</td>
<td># store string str in memory and null-terminate it.</td>
</tr>
<tr>
<td>.byte b1,...,bn</td>
<td># store n values in successive bytes of memory.</td>
</tr>
<tr>
<td>.word w1,...,wn</td>
<td># store n 32-bit quantities in successive memory words.</td>
</tr>
</tbody>
</table>

The following page has a sample program that computes 1+2+3+ ... + N
.data  #this is the data segment
prompt: .asciiz  "\n Please input a value for N = "
result: .asciiz  "The sum of the integers 1 to N = "
bye:     .asciiz  "\n Good-bye!"
.globl  main

.text  #this is the code segment
main:
    li     $v0, 4   # System call code for Print String
    la     $a0, prompt  # load address of prompt into $a0
    syscall  # Print the prompt message
    li     $v0, 5  # System call code for Read Integer
    syscall  # Read N into $v0
    blez  $v0, end   # branch to end if v0 <= 0
    li     $t0, 0
loop:   
    add    $t0, $t0, $v0  # Sum of integers in $t0
    addi   $v0, $v0, -1  # Decrement N
    bnez   $v0, loop  # branch to loop if v != 0
    li     $v0, 4   # System call code for Print String
    la     $a0, result  # load address of message into $a0
    syscall  # print the string
    li     $v0, 1  # System call code for Print Integer
    move   $a0, $t0  # move value to be printed to $a0
    syscall  # print sum of integers
    b      main  # branch to main

end:   
    li     $v0, 4   # System call code for Print String
    la     $a0, bye  # load address of message into $a0
    syscall  # print the string
    li     $v0, 10  # System call code for terminate
    syscall  # return control to system