```
!  Assembly language function to add up the number of elements in an array A.
!
!  Input:   The address of the first data item of A is given in %i0
!           The number of data items in array A is given in %i1
!
!  Variables:  Register %L0 is used as an array index.  It counts 0 to n-1.
!              Register %L1 is used to compute the address of A[i].
!              Register %L2 is used to load a data item from memory.
!              Register %L3 is used to hold the total.
!
     .align 8
     .skip 16
     .global asum

asum:
!
!   Create a new activation record with the save instruction.
!
       save %sp,-112,%sp
!
!    Initialize a counter to zero.  Also initialize the total to zero.
!
       mov %g0, %L0
       mov %g0, %L3
!
!    Use a label to define the entry point of a loop.  I.e., while (i<n) ...
!
Loop:
       cmp %L0, %i1
       bge OutOfLoop
       nop
!
!   Compute the address of A[i]
!
       smul %L0,4,%L1
       add  %i0, %L1, %L1
!
!   Load the value from memory and add it to the total
!
       ld   [%L1], %L2
       add  %L3, %L2, %L3
!
!   Add one to our counter
!
       add  %L0, 1, %L0
       ba  Loop
       nop
!
!
OutOfLoop:
!
!   Move our total into our input register %i0 to return to the caller.
!
       mov  %L3, %i0
!
!   Return
!
       jmp  %i7+8
       restore
```

```
/* -------- C Language code for a main program  -------- */

#include <stdio.h>

int asum( int A[], int n ) ;  /* Declare a function named "asum". */

int main()
{
   int A[5] ;    /* Declare our variables. */
   int n    ;
   int s    ;

   /*  Initialize the Array and n. */

   A[0] = 2 ;  A[1] = 3 ; A[2] = 5 ;  A[3] = 7 ;  A[4] = 11 ;
   n = 5 ;

   s = asum(A, n) ;

   printf("%d\n", s ) ;
}


---------------------- Sample session ----------------------
pukana% gcc -c sum.s
pukana% gcc -c main.c
pukana% gcc main.o sum.o
pukana% a.out
28
```