Karlstads universitet
Institutionen för matematik och datavetenskap
DVGA03 Datorsystemteknik

2016-12-20
Kerstin Andersson

Laboration i Assembler och Stackhantering

Syftet med laborationen är att uppöva förmågan att översätta högnivåspråk till assemblerspråk samt att använda stacken vid subrutinanrop. Vidare bör laborationen ge en djupare förståelse för hur rekursion fungerar. Detta ska åstadkommas genom översättandet av följande program som använder quicksort för att sortera en vektor i stigande ordning. Översätt koden (för hand) och implementera den i Mars. För register- och stackhantering använd den konvention som tillämpas av gcc-kompilatorn (bortse från frame pointer).

```
#include <stdio.h>
int antal = 10;
int vek[] = {4,5,2,2,1,6,7,9,5,10};

/* Funktion som grovsorterar arrayen (index a till b) */
int partition(int v[], int a, int b) {
   int pivot, lower, upper, temp;
   pivot = v[a];
   lower = a + 1;
   upper = b;
   do {
      while (v[lower] <= pivot && lower <= upper) lower = lower + 1;
      while (v[upper] > pivot && lower <= upper) upper = upper - 1;
      if (lower <= upper) {
         temp = v[lower];
         v[lower] = v[upper];
         v[upper] = temp;
         lower = lower + 1;
         upper = upper - 1;
      }
   }while (lower <= upper);
   temp = v[upper];
   v[upper] = v[a];
   v[a] = temp;
   return upper;
}

/* Rekursiv funktion som sorterar en array (index a till b) */
void quickSort(int v[], int a, int b) {
   int k;
   if (a < b) {
      k = partition(v, a, b);
      quickSort(v, a, k-1);
      quickSort(v, k+1, b);
   }
}
```

```c
/* Funktion som skriver ut elementen i en array */
void skriv(int v[], int size) {
   int i;
   printf("\n");
   for (i = 0; i < size; i++)
      printf("%d ", v[i]);
   printf("\n");
}

/* Huvudprogrammet */
int main() {
   skriv(vek, antal);
   quickSort(vek, 0, antal-1);
   skriv(vek, antal);
   return 0;
}
```

Tips: Börja med att implementera subrutinen skriv och huvudprogrammet med enbart det första subrutinanropet (till skriv). För diverse systemanrop, se tabellen nedan.

Service	System call code	Arguments	Result
print_int	$v0 = 1	$a0 = integer	
print_float	$v0 = 2	$f12 = float	
print_double	$v0 = 3	$f12 = double	
print_string	$v0 = 4	$a0 = string	
read_int	$v0 = 5		integer (in $v0)
read_float	$v0 = 6		float (in $f0)
read_double	$v0 = 7		double (in $f0)
read_string	$v0 = 8	$a0 = buffer, $a1 = length	
sbrk	$v0 = 9	$a0 = amount	address (in $v0)
exit	$v0 = 10		
print_char	$v0 = 11	$a0 = char	
read_char	$v0 = 12		char (in $v0)
open	$v0 = 13	$a0 = filename (string), $a1 = flags, $a2 = mode	file descriptor (in $a0)
read	$v0 = 14	$a0 = file descriptor, $a1 = buffer, $a2 = length	num chars read (in $a0)
write	$v0 = 15	$a0 = file descriptor, $a1 = buffer, $a2 = length	num chars written (in $a0)
close	$v0 = 16	$a0 = file descriptor	
exit2	$v0 = 17	$a0 = result	

Från Patterson & Hennessy (A.9)