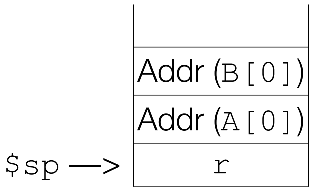
►Solution◄

Question 1: (15 points)
For this question, assume that:

- p, q, i, j are 32-bit integers whose values are stored in $s0, $s1, $s2, and $s3, respectively.
- A and B are arrays of integers.
- r is a pointer declared as int *r.
- r, the base address of array A, and the base address of array B are all in the stack frame of the current function, as shown below



For each of the C statements below, give the translation into MIPS. Do not use pseudo-instructions in your code. Clearly label which MIPS instructions are for which statement.

a. (5 points) q = *r

Solution:

```
lw      $t0, 0($sp)      # $t0 <-- Address r points to
lw      $s1, 0($t0)      # $q <-- Value at address in $t0
```

b. (5 points) B[i] = A[j]

Solution:

```
sll      $t1, $s3, 2      # $t1 <-- 4*j
lw       $t0, 4($sp)      # $t0 <-- Addr(A[0])
add      $t1, $t0, $t1    # $t2 <-- Addr(A[j])
lw       $t0, 0($t1)      # $t0 <-- A[j]
sll      $t1, $s2, 2      # $t1 <-- 4*i
lw       $t2, 8($sp)      # $t2 <-- Addr (B[0])
add      $t2, $t2, $t1    # $t2 <-- Addr(B[i])
sw       $t0, 0($t2)      # $t0 <-- Address in $t2
```

c. (5 points) p = q + A[B[j]]

Solution:

```
sll      $t1, $s3, 2      # $t1 <-- 4*j
lw       $t0, 8($sp)      # $t0 <-- Addr(B[0])
add      $t1, $t0, $t1    # $t1 <-- Addr(B[j])
lw       $t0, 0($t1)      # $t0 <-- B[j]
sll      $t0, $t0, 2      # $t0 <-- 4*B[j]
lw       $t1, 4($sp)      # $t1 <-- Addr (A[0])
add      $t1, $t0, $t1    # $t1 <-- Addr(A[B[j]])
lw       $t0, 0($t1)     # $t0 <-- A[B[j]]
add      $s0, $s1, $t0    # $s0 <-- q + A[B[j]]
```