Bresia Prudente, bprude2
3/8/2013

4.45

A.

```
#include <stdio.h>
#include <stdlib.h>

void bubble_a(int *data, int count)
{
        int i, last;

        for(last = count-1; last > 0; last--)
        {
                for(i = 0; i < last; i++)
                {
                  /* We can evalute x[i] as *(x+i) */
                        if(*(data+i+1) < *(data+i))
                        {

                                /* Swap adjacent elements */
                                int t = *(data+i+1);
                                data[i+1] = *(data+i);
                                *(data+i) = t;

                        }//end if

                }//end for

        }//end for

}//end void bubble_a

int main()
{

      /* Test the array by adding some elements */
      int arr[5] = {3, 8, 1, 2, 4};

      bubble_a(arr, 5);
      int x;

      /* This will print each element of the array */
      for(x = 0; x < 5; x++)
      {

            printf("%d\n", arr[x]);

      }//end for
      return 0;

}//end main
```

B.

```
.globl bubble_a
        .type   bubble_a, @function
_bubble_a:
.LFB5:
        pushl   %rbp
.LCFI0:
        rrmovl  %rsp, %rbp
.LCFI1:
        rmmovl  %rdi, -24(%rbp)
        rmmovl  %esi, -28(%rbp)
        mrmovl  -28(%rbp), %eax
        subl    $1, %eax
        rmmovl  %eax, -8(%rbp)
        jmp     .L2
.L3:
        irmovl  $0, -12(%rbp)
        jmp     .L4
.L5:
        mrmovl  -12(%rbp), %eax
        cltq
        sall    $2, %rax
        addl    -24(%rbp), %rax
        addl    $4, %rax
        mrmovl  (%rax), %edx
        mrmovl  -12(%rbp), %eax
        cltq
        sall    $2, %rax
        addl    -24(%rbp), %rax
        mrmovl  (%rax), %eax
        pushl   %edx
        subl    %eax, %edx
        popl    %ecx
        jge     .L6
        rmmovl  -12(%rbp), %eax
        cltq
        sall    $2, %rax
        addl    -24(%rbp), %rax
        addl    $4, %rax
        mrmovl  (%rax), %eax
        rmmovl  %eax, -4(%rbp)
        mrmovl  -24(%rbp), %rdx
        addl    $4, %rdx
        mrmovl  -12(%rbp), %eax
        cltq
        sall    $2, %rax
        addl    %rax, %rdx
        mrmovl  -12(%rbp), %eax
        cltq
        sall    $2, %rax
        addl    -24(%rbp), %rax
        mrmovl  (%rax), %eax
        rmmovl  %eax, (%rdx)
```

```
        mrmovl  -12(%rbp), %eax
        cltq
        sall    $2, %rax
        rrmovl  %rax, %rdx
        addl    -24(%rbp), %rdx
        mrmovl  -4(%rbp), %eax
        rmmovl  %eax, (%rdx)
.L6:
        addl    $1, -12(%rbp)
.L4:
        movl    -12(%rbp), %eax
        pushl   %eax
        subl    -8(%rbp), %eax
        popl    %eax
        jl      .L5
        subl    $1, -8(%rbp)
.L2:
        pushl   -8(%rbp)
        subl    $0
        popl    -8(%rbp)
        jg      .L3
        leave
        ret
        .cstring
.LC0:
        .ascii  "%d\n"
        .text
.globl main
_main:
.LFB6:
        pushl   %rbp
.LCFI2:
        rrmovl  %rsp, %rbp
.LCFI3:
        subl    $32, %rsp
.LCFI4:
        immovl  $3, -32(%rbp)
        immovl  $8, -28(%rbp)
        immovl  $1, -24(%rbp)
        immovl  $2, -20(%rbp)
        immovl  $4, -16(%rbp)
        leaq    -32(%rbp), %rdi
        irmovl  $5, %esi
        call    bubble_a
        immovl  $0, -4(%rbp)
        jmp     .L12
.L13:
        mrmovl  -4(%rbp), %eax
        cltq
        movl    -32(%rbp,%rax,4), %esi
        mrmovl  $.LC0, %edi
        irmovl  $0, %eax
        call    printf
        addl    $1, -4(%rbp)
.L12:
        pushl   -4(%rbp)
        subl    $4, -4(%rbp)
```

```
popl    -4(%rbp)

jle     .L13
irmovl  $0, %eax
leave
ret
```

4.47

Fetch:

icode : ifunc ←M_1[PC]
rA : rB ←M_1[PC+1]
valC ←M_4[PC+2]
valP ←PC+6

Decode:

valB ←R[rB]

Execute:

valE ←valB+valC

Memory:

N/A

Write Back:

R[rB] ←valE

PC update:

PC ←valP