Solution for assignment6

1. (I am not sure if my sort.ys works, so I copy my y86 codes here in order to check in a simulator.)

```
.pos 0
Init:
   irmovl Stack, %ebp
   irmovl Stack, %esp
   call Main
   halt
.pos 0x100
Stack:
array:
   .long 0x0005
   .long 0x0002
   .long 0x0001
   .long 0x0004
   .long 0x0003
   .long 0x0006
   .long 0x0008
   .long 0x0007
   .long 0x0009
   .long 0x000a


Main:
   pushl %ebp
   rrmovl %esp,%ebp
   irmovl array, %edi //addre of first data
   irmovl $10, %esi  //size
   irmovl $1, %eax
   subl %eax, %esi //last index
   call Sort
   rrmovl %ebp, %esp
   popl %ebp
   ret

.pos 0x200
Sort:
   pushl %ebp
   rrmovl %esp,%ebp
   pushl %ebx
   pushl %esi
```

```
loop:
   irmovl $0, %edx
   subl %edx, %esi
   jle End   //if last index <0, end
   call Getmax
   addl %eax, %eax
   addl %eax, %eax //4*eax, get the max position
   addl %edi, %eax //address of it

   rrmovl %esi, %ecx //get copy of esi
   addl %ecx, %ecx
   addl %ecx, %ecx //4*ecx, last position
   addl %edi, %ecx //address of it

   mrmovl (%eax),%edx //get max value inside array
   mrmovl (%ecx), %ebx //get the value at last position
   rmmovl %edx, (%ecx)
   rmmovl %ebx, (%eax) //swap them

   irmovl $1, %ecx
   subl %ecx, %esi //index of last -1
   jmp loop

End:
   popl %esi
   popl %ebx
   rrmovl %ebp, %esp
   popl %ebp
   ret

.pos 0x300

Getmax:
   pushl %ebp
   rrmovl %esp,%ebp
   pushl %edi    //the first addr of array
   pushl %esi    //size
   pushl %ebx

   rrmovl %esi, %eax
   addl %eax, %eax
   addl %eax, %eax  //size*4
   addl %edi, %eax  //get the addr
   rrmovl %eax, %ebx //copy to get the addr
   mrmovl (%ebx), %ebx  //deference ebx to get value a[n]
   rrmovl %esi, %edx

while:
   xorl %eax, %eax
   subl %eax, %esi //set condition
   jle Done

   irmovl $1, %eax
   subl %eax, %esi   //last index -1
   rrmovl %esi, %eax
   addl %eax, %eax
   addl %eax, %eax // number *4
   addl %edi, %eax //get addr
   mrmovl (%eax), %eax //dereference

   rrmovl %eax, %ecx
   subl %ebx, %eax  //eax = max-x
   cmovg %ecx, %ebx //compare max, x
   cmovg %esi, %edx //compare position,n

   jmp while

Done:
   rrmovl %edx, %eax
   popl %ebx
   popl %esi
   popl %edi
   rrmovl %ebp, %esp
   popl %ebp
   ret
```

2.

```
addl  %edx, %eax
mrmovl  0(%ecx), %edx
addl %edx, %eax
```

with forwarding:

```
addl  %edx, %eax
mrmovl  0(%ecx), %edx
addl %edx, %eax
```

F	D	E	M	W		
	F	D	E	M	W	
		F	D	E	M	W

No need pipeline stalls or bubbles with forwarding.

Without forwarding:

```
addl  %edx, %eax
mrmovl  0(%ecx), %edx
nop
nop
nop
addl %edx, %eax
```

F	D	E	M	W					
	F	D	E	M	W				
		F	D	E	M	W			
			F	D	E	M	W		
				F	D	E	M	W	
					F	D	E	M	W

Need 3 pipeline stalls or bubbles without forwarding.