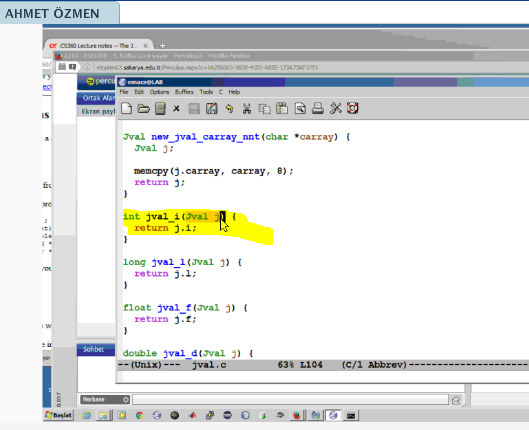
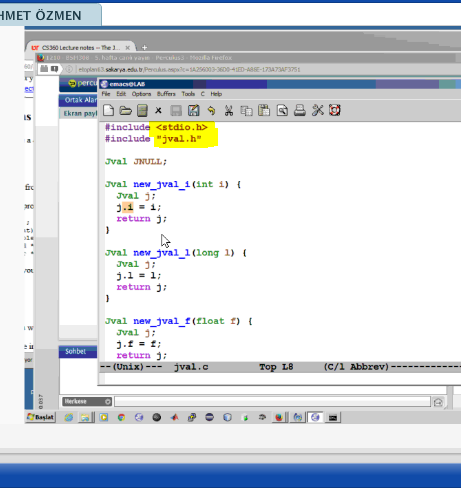
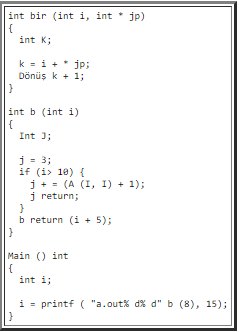
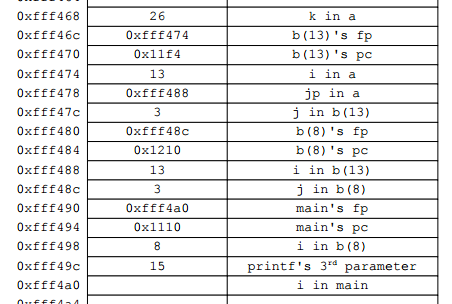
Basit C program yazdırma soracak.



Yukarda 2 satır 2. Kelimeyi bulan programı yazınız.



C de yazılmış hali solda makine diline çevrilmiş hali

**CS360 Lab #5 -- Assembler**

* [**James S. Plank**](http://www.cs.utk.edu/~plank)
* [**CS360**](http://www.cs.utk.edu/~plank/plank/classes/cs360/)

This is not a lab that you hand in. Instead, its goal is to help you reinforce the lecture notes on assembler, and to study for the exam. For that reason, I have included the answers with the lab. It is completely fair game for you to consult the TA's to help you with this material.

Additional study material:

* Question 1 from the 2011 Midterm Exam.
* Questions 3 and 4 from the 2004 Midterm Exam.
* Question 1 from the 2003 Midterm Exam. Maybe I should solicit money from the students to prevent me from asking a question like this again...

Do not "optimize" the assembler. Give me the simple yet inefficient assembler that the compiler would return.

**Question 1**

Derive the assembler for the following procedure.

int b(int j, int k)

{

int i;

i = j;

while (i > 0) {

j = (i \* j) + (k \* 3);

i--;

}

return j;

}

The answer is [here](http://web.eecs.utk.edu/~plank/plank/classes/cs360/360/labs/lab6/a1.html).

Çözümü:

b:

push #4 / Allocate i

st %r2 -> [sp]-- / Spill r2 since you'll use it

ld [fp+12] -> %r0 / i = j

st %r0 -> [fp]

l1:

ld [fp] -> %r0 / while (i > 0)

cmp %r0, %g0

ble l2

ld [fp] -> %r0 / j = (i\*j) + (k\*3)

ld [fp+12] -> %r1

mul %r0, %r1 -> %r0

ld [fp+16] -> %r1

mov #3 -> %r2

mul %r1, %r2 -> %r1

add %r0, %r1 -> %r0

st %r0 -> [fp+12]

ld [fp] -> %r0 / i--

add %r0, %gm1 -> %r0

st %r0 -> [fp]

b l1

l2:

ld ++[sp] -> %r2 / Unspill r2

ld [fp+12] -> %r0 / return j

ret

**Question 2**

Derive the assembler for the following procedure. You may assume that NULL is equal to zero.

int f2(int \*x)

{

int j;

if (x == NULL) return 37;

j = \*x \* 2;

return j;

}

The answer is [here](http://web.eecs.utk.edu/~plank/plank/classes/cs360/360/labs/lab6/a2.html).

Çözümü:

f2:

push #4 / Allocate j

ld [fp+12] -> %r0

cmp %r0, %g0 / if (x == NULL)

bne l1

mov #37 -> %r0 / return 37

ret

l1:

ld [fp+12] -> %r0 / j = \*x \* 2

ld [r0] -> %r0

mov #2 -> %r1

mul %r0, %r1 -> %r0

st %r0 -> [fp]

ld [fp] -> %r0 / return j

ret

**Question 3**

Derive the assembler for the following program. In terms of grading, this would be worth three times Question 1 or Question 2.

int c(int i, int j)

{

if (i <= 0) return j+1;

return a(j, i-3)+1;

}

int a(int i, int j)

{

if (j <= 0) return i+1;

return c(i-1, j)+c(i-1, j-1);

}

main(int argc, char \*\*argv)

{

int i;

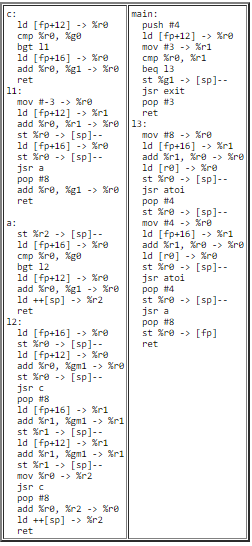
if (argc != 3) { exit(1); }

i = a(atoi(argv[1]), atoi(argv[2]));

}

The answer is [here](http://web.eecs.utk.edu/~plank/plank/classes/cs360/360/labs/lab6/a3.html).

Çözümü:



**Question 4**

Below is a snapshot of the stack when the program in Question 3 is executing. As in the "fact" example in [Assembler Lecture #3](http://www.cs.utk.edu/~plank/plank/classes/cs360/360/notes/Assembler3/lecture.html), label each byte of the stack, and show where we are in the execution of the program (don't just label the PC -- show where we are in the call stack). You may assume that **main** starts at label 0x10a0, **a** starts at label 0x1040, and **c** starts at label 0x1000. Here's the Open Office of the picture if you want: [**q3-unlabeled.odg**](http://web.eecs.utk.edu/~plank/plank/classes/cs360/360/labs/lab6/q3-unlabeled.odg).

|  |
| --- |
| http://web.eecs.utk.edu/~plank/plank/classes/cs360/360/labs/lab6/q3-unlabeled.png |

The answer is [here](http://web.eecs.utk.edu/~plank/plank/classes/cs360/360/labs/lab6/q4-labeled.pdf).