# 一个简单的问题：编写迭代求和的选择

假设有一个问题，需要求出 数组a中下标m到n的和，如果要求写汇编代码，你会怎么写？

显然，无论如何，你都会开始搜索记忆中关于汇编的知识，然后想到大概使用循环结构能够完成这个问题，但具体到细节，你还不太清楚，让我们开始构建一个汇编代码 吧！

首先，需要能够表示输入参数中的 数组a 下标m和n的符号，假设在汇编语言中定义数组和符号如下：

a:

.long 2,3,5,8,13,21

size = 6

m = 2

n = 5

则最后我们输出的结果应当是 5 + 8 + 13 + 21 = 47

现在开始编写汇编代码：

mov $a,%ebx #ebx作为指向当前所要加的数的指针

add $m,%ebx

xor %eax,%eax #eax将作为累加器

mov $n,%ecx

sub $m,%ecx #ecx = n – m作为计数器

.add\_one\_integer:

add (%ebx),%eax #累加

add $4,%ebx #ebx指向下一个数

loop .add\_one\_integer #loop指令每执行一次都会将ecx减1，然后检查其是否为0，如果为0则不跳转，不为零则跳转

#到这里，eax就是我们所要的求和值。

############################经过挣扎，我写出的代码如下

#include <cstdio>

int a[]={2,3,5,8,13,21};

int m=1;

int n=5;

int main()

{

int eax;

/\*

printf("sizeof(int)= %d\n",sizeof(int));

int s=0;

s+=a[0];

int j=5;

s+=a[j];

\*/

\_\_asm\_\_ \_\_volatile\_\_(

"xor %%rbx,%%rbx \n\t"

"xor %%rax,%%rax \n\t"

"mov $a,%%rbx #ebxΣ╜£Σ╕║µîçσÉæσ╜ôσëìµëÇΦªüσèáτÜäµò░τÜäµîçΘÆê\n\t"

"mov m(%%rip),%%eax \n\t"

"shl $2,%%eax \n\t"

"add %%rax,%%rbx\n\t"

"xor %%rcx,%%rcx \n\t"

"mov n(%%rip),%%ecx\n\t"

"mov m(%%rip),%%eax \n\t"

"sub %%rax,%%rcx #ecx = n ΓÇô mΣ╜£Σ╕║Φ«íµò░σÖ¿\n\t"

"xor %%rax,%%rax #eaxσ░åΣ╜£Σ╕║τ┤»σèáσÖ¿\n\t"

".add\_one\_integer:\n\t"

"add (%%rbx),%%eax #τ┤»σèá\n\t"

"#mov (,%%rbx),%%eax \n\t"

"add $4,%%rbx #ebxµîçσÉæΣ╕ïΣ╕ÇΣ╕¬µò░\n\t"

"loop .add\_one\_integer\n\t"

:"=a"(eax)

:

:"rcx"

);

printf("rax %d\n",eax);

/\*

\_\_asm\_\_("popq %rax \n\t");

\*/

}