八皇后问题

—— 汇编程序设计

使用工具：MARS

作业内容　　编写一个计算八皇后问题的程序

1．计算的规则和任务

国际象棋中的皇后可以吃掉与它在同一行、同一列、同一对角线上的棋子。八皇后问题，即在8×8 的国际象棋棋盘上放置8 个皇后，要求任意两个皇后不能在同一行、同一列或同一条对角线上。求出如此放置方法的种数。

一种解决问题的思路是一行放置皇后，如果当前放置的皇后与前面的皇后不存在冲突时，则继续摆下一个皇后，否则跳到上一个皇后，重新摆置。

编写一个***MIPS*** 软件，要求和下面的程序功能相同，尤其请注意其中的键盘输入和打印功能也需要实现。

2．函数的*C* 语言形式描述

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

int Site[8];
int Queen(int n, int QUEENS, int);
int Valid(int n);

void main()
{
    int m;
    int iCount = 0;
    int n ;
    printf("Eight Queen problems, entering the number of queens:") ;
    scanf("%d" , &n) ;
    m=Queen(0,n , iCount);
    printf("%d\n",m);
```

```
    return ;
}

/*-----------------Queen：递归放置第n 个皇后---------------*/
int Queen(int n, int QUEENS, int iCount)
{
    int i;
    if(n == QUEENS)
    {
        iCount=iCount+1;
        return iCount;
    }
   for(i = 1 ; i <= QUEENS ; i++)
   {
        Site[n] = i;
        /*---------------Valid：判断放置第n 个皇后时是否无冲突---------------*/
        if(Valid(n))
        iCount = Queen(n + 1,QUEENS, iCount);
   }
   return iCount;
}

/*------Valid：判断第n 个皇后放上去之后，是否合法，即是否无冲突。------*/
int Valid(int n)
{
    int i;
    for(i = 0 ; i < n ; i++)
    {
        if(Site[i] == Site[n])
        return 0;
        if(abs(Site[i] - Site[n]) == (n - i))
        return 0;
    }
    return 1;
}
```

3. 作业说明

(1) n, QUEENS　均为整数，求解N皇后问题时，调用Queen(0,N)

(2) 当N=8 时，八皇后的解为92 种，可以用这个结果来测试程序的正确性

(3) 八皇后问题也可以用非递归形式实现，但是为了让大家对于编写递归程序有一个比较好的锻炼，要求程序用递归方式实现

(4) 参考资料

教材：计算机硬件对过程的支持

作业：栈和过程

确定如何用分支语句来翻译C语言中的if判断语句和for　　语句等；参阅讲义和教材掌握利用MIPS汇编语言如何进行过程调用，熟悉栈的使用以及递归程序设计；

(5) Tips:

a) 由于变量不多，建议你可以为每一个变量（参数、临时变量）都分配一个寄存器，在手边记录下来哪个寄存器是表示哪个变量的，这样编程会比较简单，lw　和sw语句很少；

b) 可以参考一下教材后面伪指令的用法（一条伪指令往往对应了两条甚至以上的MIPS　指令）；这会方便程序设计。

常用的伪指令：

la $a0 , __begincheck

la 是一种常见伪指令，可以将任意地址处的一个标签地址加载到某寄存器中，一般翻译为lui　和ori　两条指令：

li $v0 , 5

li 是load immediate的意思，给寄存器赋值。

Syscall　的用法：

代码	系统调用	参数	结果
1	print integer	$a0	
2	print float	$f12	
3	print double	$f12	

4	print string	$a0	
5	read integer		integer in $v0
6	read float		float in $f0
7	read double		double in $f0
8	read string	$a0=buffer, $a1=length	
9	sbrk	$a0=amount	address in $v0
10	exit	$a0=result	
11	print char	$a0	
12	read char		char in $v0
13	open	$a0=file name(string), $a1=flags, $a2=mode	file descriptor (fd) in $v0
14	read	$a0 =fd, $a1=buffer, $a2=length	num chars read in $v0
15	write	$a0 =fd, $a1=buffer, $a2=length	num chars write in $v0
16	close	$a0 =fd	
17	exit2	$a0=result	

例如要打印一字符串:

```
li $v0 , 4
la $a0 , __string
syscall
.data
__string:
.asciiz  hello!
```

而要从键盘输入一个数字，则可以用第 5 号功能的 syscall。