附件３

西安电子科技大学

微机原理 课程第四章作业报告

名称 奇数的和

人工智能 学院 2120011 班

姓名 游霄童 学号 21009200158

同作者 无

日期 2023 年 10 月 17 日

成　绩

奇数的和

姓名：游霄童 学号：21009200158

created：2023/10/17

问题描述

用同余法产生200 个小于256 的伪随机数，统计其中奇数的个数，并计算所有奇数的和，将奇数个数存入名为CNT 的字节单元，和存入名为SUMODD 的字存储单元中。用完整的段定义语句编写出实现这一功能的汇编语言源程序。

一、基本原理及步骤（或方案设计及理论计算）

1.同余数

同余法是一种用于生成伪随机数的方法。它基于一个递归的数学公式生成序列，看起来像是随机的数字序列。同余法的基本思想是通过递推关系式生成一系列的整数，然后通过对这些整数进行适当的处理，得到在指定范围内均匀分布的伪随机数。步骤如下：

选择初始值：选择一个初始值作为种子数。

选择同余公式参数：选择合适的 (a)、(c) 和 (m) 值，满足 (0 < a, c < m)，通常选择素数 (m)。

递推生成随机数：使用同余法公式

$$X_{n+1} = (aX_n + c) \mod m$$

递推生成一系列的整数。

归一化处理：将得到的整数除以 (m)，得到 [0, 1) 范围内的浮点数，即为所求的伪随机数。

在上述实验中，使用同余法生成了200个小于256的伪随机数。首先，选择了初始种子数为0，同余法的参数 (a) 为11，(c) 为3。然后，通过循环运算，利用同余法公式生成了200个伪随机数。在生成过程中，判断每个数是否为奇数，统计奇数的个数，并计算所有奇数的和。

这个实验的核心在于了解同余法的原理，正确选择参数以及使用循环结构生成一系列的伪随机数。通过不断递推和判断，可以得到满足要求的伪随机数序列。

2.运用8086指令完成上述随机数生成及求和

初始化段定义和变量：定义了堆栈段（大小为100H字节），数据段和代码段。在数据段中定义了用于存储结果的变量：CNT（奇数的个数）和SUMODD（奇数的和），以及用于同余法计算的变量VAR1、VAR2和VAR3。**初始化寄存器和循环计数器**：将数据段地址加载到DS寄存器。清零AX、BX、CX和DX寄存器。将VAR1的值加载到AL寄存器，作为循环计数器。将VAR2的值加载到BL寄存器，用于同余法计算。将VAR3的值加载到CL寄存器，控制循环次

数。外层循环（SUM标签）：将外层循环次数（10次）压入堆栈，用于内层循环的控制。进入内层循环（TONGYU标签）。**内层循环（TONGYU标签）**：使用同余法公式

$$X_{n+1} = (aX_n + c) \mod m$$

计算AX的值。判断计算结果是否为奇数。如果是奇数，将其加到SUMODD中，并将奇数的个数CNT加1。**循环结束和输出**：判断内层循环是否结束，如果没有，继续循环，否则退出内层循环。判断外层循环是否结束，如果没有，继续循环，否则退出外层循环。将奇数的个数CNT和奇数的总和SUMODD存入相应的变量。使用DOS中断21H退出程序运行。

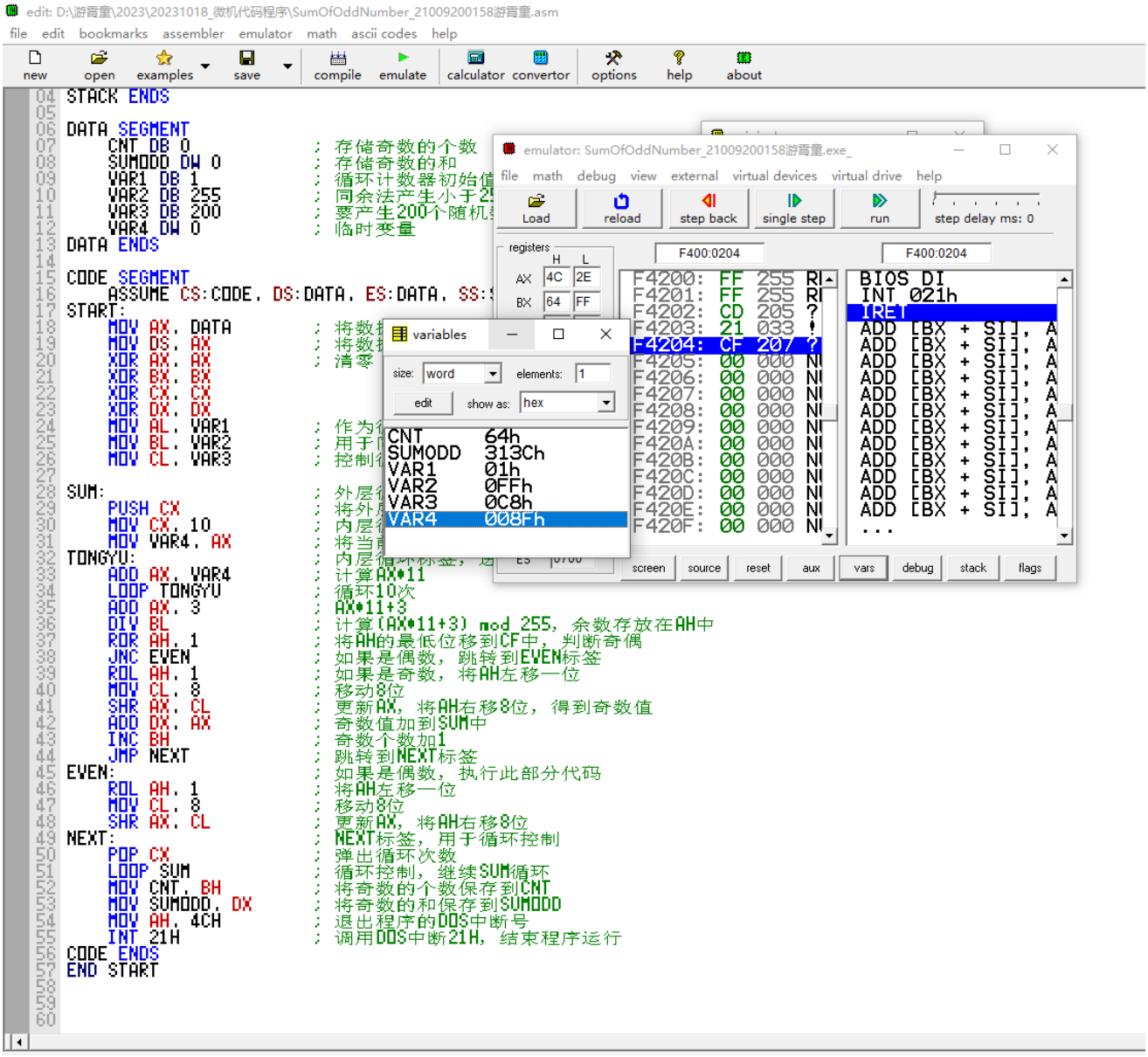
这个程序的核心在于使用同余法生成伪随机数，判断每个数是否为奇数，并统计奇数的个数和总和。循环结构使得上述操作可以重复执行，直到生成足够的随机数。最终，程序将奇数的个数和总和保存在CNT和SUMODD变量中，并通过DOS中断退出程序。

二、代码

```
STACK SEGMENT STACK'STACK'
    DW 100H DUP(?)      ; 定义堆栈段，大小为100H字节
TOP LABEL WORD
STACK ENDS

DATA SEGMENT
    CNT DB 0            ; 存储奇数的个数
    SUMODD DW 0         ; 存储奇数的和
    VAR1 DB 1           ; 循环计数器初始值
    VAR2 DB 255         ; 同余法产生小于256的随机数：AX=(AX*11+3) mod 255
    VAR3 DB 200         ; 要产生200个随机数，循环200次
    VAR4 DW 0           ; 临时变量
DATA ENDS

CODE SEGMENT
    ASSUME CS:CODE, DS:DATA, ES:DATA, SS:STACK
START:
    MOV AX, DATA        ; 将数据段地址加载到AX寄存器
    MOV DS, AX          ; 将数据段地址复制到DS寄存器
    XOR AX, AX          ; 清零
    XOR BX, BX
    XOR CX, CX
    XOR DX, DX
    MOV AL, VAR1        ; 作为循环计数器
    MOV BL, VAR2        ; 用于同余法计算
    MOV CL, VAR3        ; 控制循环次数

SUM:                    ; 外层循环标签，计算奇数的和
    PUSH CX             ; 将外层循环次数压入堆栈
    MOV CX, 10          ; 内层循环次数
    MOV VAR4, AX        ; 将当前计数器值保存到VAR4
TONGYU:                 ; 内层循环标签，进行同余法计算
    ADD AX, VAR4        ; 计算AX*11
```

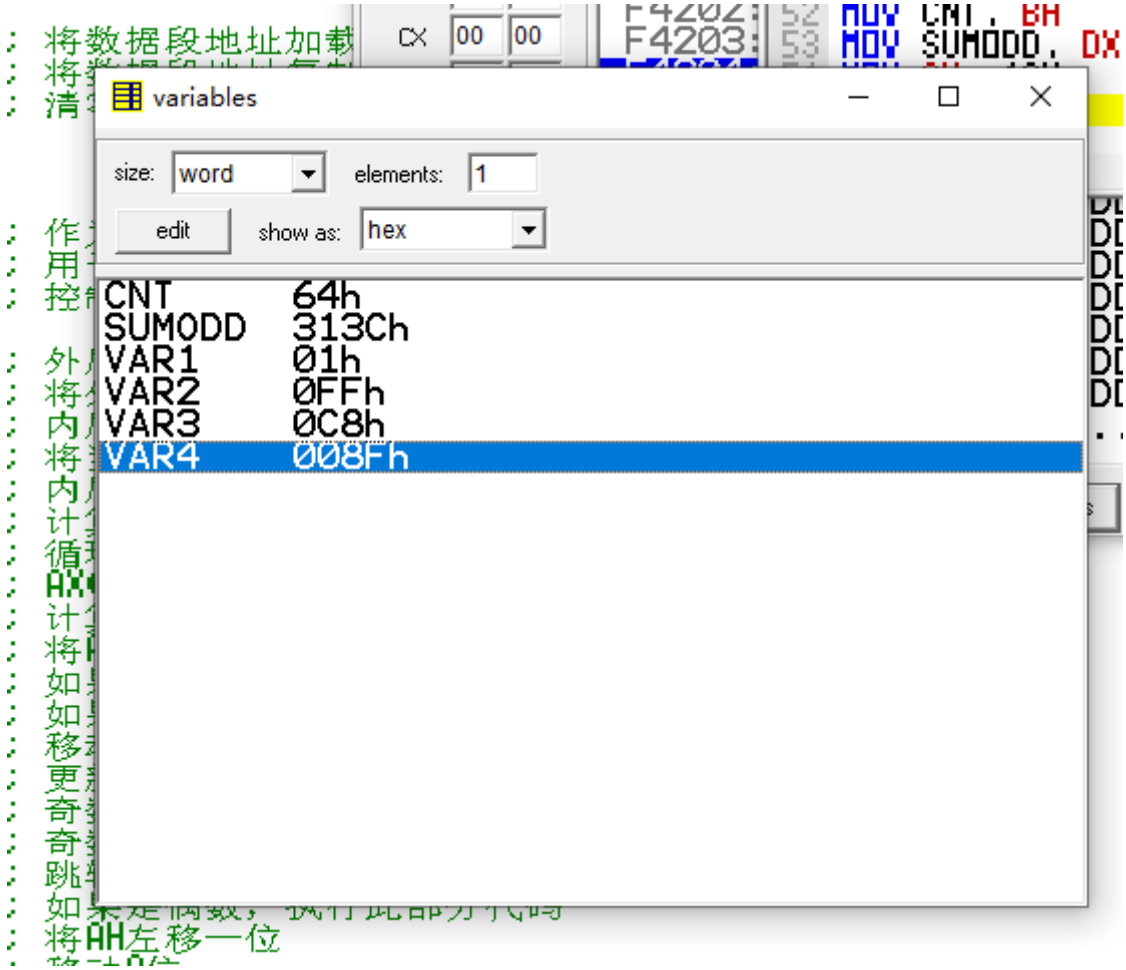
```
    LOOP TONGYU          ; 循环10次
    ADD AX, 3            ; AX*11+3
    DIV BL               ; 计算(AX*11+3) mod 255，余数存放在AH中
    ROR AH, 1            ; 将AH的最低位移到CF中，判断奇偶
    JNC EVEN             ; 如果是偶数，跳转到EVEN标签
    ROL AH, 1            ; 如果是奇数，将AH左移一位
    MOV CL, 8            ; 移动8位
    SHR AX, CL           ; 更新AX，将AH右移8位，得到奇数值
    ADD DX, AX           ; 奇数值加到SUM中
    INC BH               ; 奇数个数加1
    JMP NEXT             ; 跳转到NEXT标签
EVEN:                    ; 如果是偶数，执行此部分代码
    ROL AH, 1            ; 将AH左移一位
    MOV CL, 8            ; 移动8位
    SHR AX, CL           ; 更新AX，将AH右移8位
NEXT:                    ; NEXT标签，用于循环控制
    POP CX               ; 弹出循环次数
    LOOP SUM             ; 循环控制，继续SUM循环
    MOV CNT, BH          ; 将奇数的个数保存到CNT
    MOV SUMODD, DX       ; 将奇数的和保存到SUMODD
    MOV AH, 4CH          ; 退出程序的DOS中断号
    INT 21H              ; 调用DOS中断21H，结束程序运行
CODE ENDS
END START
```

三、结果与分析




```
STACK ENDS

DATA SEGMENT
    CNT DB 0                ; 存储奇数的个数
    SUMODD DW 0             ; 存储奇数的和
    VAR1 DB 1               ; 循环计数器初始值
    VAR2 DB 255             ; 同余法产生小于25
    VAR3 DB 200             ; 要产生200个随机数
    VAR4 DW 0               ; 临时变量
DATA ENDS

CODE SEGMENT
    ASSUME CS:CODE, DS:DATA, ES:DATA, SS:
START:
    MOV AX, DATA            ; 将数
    MOV DS, AX              ; 将数
    XOR AX, AX              ; 清零
    XOR BX, BX
    XOR CX, CX
    XOR DX, DX
    MOV AL, VAR1            ; 作为
    MOV BL, VAR2            ; 用于
    MOV CL, VAR3            ; 控制

SUM:                        ; 外层
    PUSH CX                 ; 将外
    MOV CX, 10              ; 内层
    MOV VAR4, AX            ; 将当
TONGYU:                     ; 内层循环标签，
    ADD AX, VAR4            ; 计算AX*11
    LOOP TONGYU             ; 循环10次
    ADD AX, 3               ; AX*11+3
    DIV BL                  ; 计算(AX*11+3) mod 255，余数存放在AH中
    ROR AH, 1               ; 将AH的最低位移到CF中，判断奇偶
    JNC EVEN                ; 如果是偶数，跳转到EVEN标签
    ROL AH, 1               ; 如果是奇数，将AH左移一位
    MOV CL, 8               ; 移动8位
    SHR AX, CL              ; 更新AX，将AH右移8位，得到奇数值
    ADD DX, AX              ; 奇数值加到SUM中
    INC BH                  ; 奇数个数加1
    JMP NEXT                ; 跳转到NEXT标签
EVEN:                       ; 如果是偶数，执行此部分代码
    ROL AH, 1               ; 将AH左移一位
    MOV CL, 8               ; 移动8位
    SHR AX, CL              ; 更新AX，将AH右移8位
NEXT:                       ; NEXT标签，用于循环控制
    POP CX                  ; 弹出循环次数
    LOOP SUM                ; 循环控制，继续SUM循环
    MOV CNT, BH             ; 将奇数的个数保存到CNT
    MOV SUMODD, DX          ; 将奇数的和保存到SUMODD
    MOV AH, 4CH             ; 退出程序的DOS中断号
    INT 21H                 ; 调用DOS中断21H，结束程序运行
CODE ENDS
END START
```

其中：



得到结果 CNT = 64H ，即200个数中产生了100个奇数，且和为SUMODD=313CH，即12684

在上述程序中，通过同余法生成了200个小于256的伪随机数，统计了其中的奇数个数并计算了所有奇数的和。CNT中存储奇数的个数，SUMODD中存储奇数的和。根据输入的种子值和同余法的参数，程序生成的伪随机数序列将在每次执行时有所不同。

四、收获及心得体会

通过这个实验，我深入了解了同余法这种伪随机数生成的方法。我学会了如何使用汇编语言编写相应的代码来实现这一算法，并且理解了奇数统计和求和的基本原理。在编写过程中，我对寄存器的使用和逻辑判断有了更深入的认识，也锻炼了编写结构化程序的能力。这次实验使我更加熟悉了汇编语言的语法和逻辑，为我今后深入学习微机原理与接口技术提供了良好的基础。