

# 同济大学

## 汇编语言程序设计 (上机大作业报告)

学 号: \_\_\_\_\_

姓 名: \_\_\_\_\_

专 业: 计算机科学与技术

任课老师: 张光长

# 一、设计题目

某学生利用暑期到快递公司打工，该公司以底薪加计件工资的形式，并以周为结算周期给实习学生发工资。具体计薪办法是：实习学生一周工作 6 天，每周基本工资 600 元，每天送快递 80 件为基本要求，每天多送 1 件增加 1.5 元，每天不足 80 件则每少 1 件扣 1.2 元。某同学某周内各天的快递量分别为 102, 90, 67, 89, 98, 125。编程计算该实习学生本周能领到多少工资？

# 二、设计说明

## 1. 执行流程

程序接受从键盘输入的一周各天的快递量，首先判断是否为合法的数字输入。每天的基本工资为  $600/6=100$  元，并且以 80 件为界限，少于 80 件每件扣除 1.2 元，多于 80 件则每件增加 1.5 元，将每日的实际工资进行累加，计算求出周工资，将周工资保留一位小数在屏幕上进行显示。并且给出相应的提示信息与错误处理信息。

## 2. 流程设计

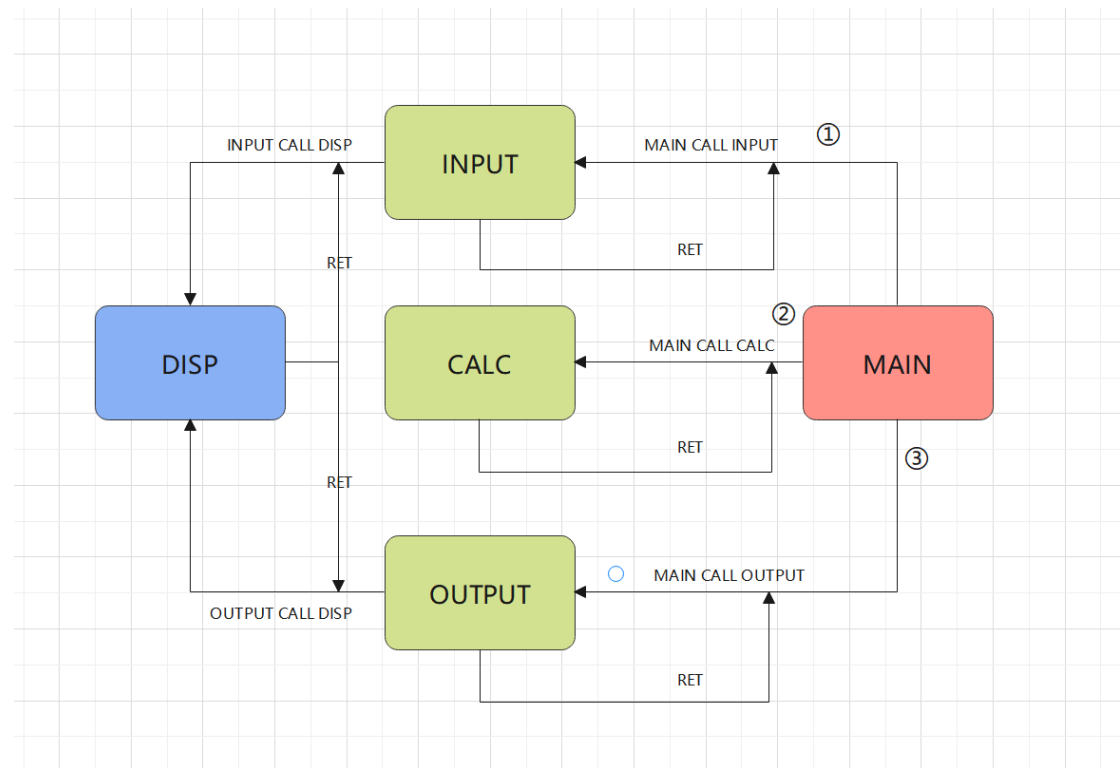
整个程序流程分为：MAIN、INPUT、CUL、OUTPUT、DISP 五个子程序，其中 MAIN 进行对 INPUT、CALC、OUTPUT 的调用完成整体工作。CALC 实现对日工资的计算和周工资的累加。OUTPUT 完成输出结果工作。DISP 实现屏幕上显示十进制数字，INPUT 和 OUTPUT 均调用 DISP。

## 3. 子程序功能及参数传递

子程序名称	功能说明	入口参数	出口参数
MAIN	主程序，通过调用 INPUT 、 CALC 、 OUTPUT 完成整体工作。	无	无
INPUT	完成输入功能：包括输入提示信息、通过键盘输入读取数据、判断所输入数据的合理性（如非法，给出错误提示）。	无	NUM（数组，存放每天快递量）
CALC	完成计算功能：对 NUM 中存储数据，按照题目原则计算出每日工资，并累加求出周工资。	BX（NUM 首地址）	ANS（结果，即所求周工资）

OUTPUT	完成输出功能：将ANS 保留1 位小数，打印到屏幕上。	ANS	无
DISP	完成二进制数据以十进制形式打印到屏幕上。	AX（待打印数据）	无

#### 4. 程序框图



#### 5. 子程序说明及其流程图

##### 5.1 MAIN

###### 1) 说明

主程序。首先调用 INPUT 子程序读取键盘输入的数据，并进行非法信息检测，将各天快递量存储于一个连续数组。然后调用 CALC 子程序，根据快递量计算对应的日工资、周工资，将周工资的计算结果存放于变量中。最后调用 OUTPUT 子程序，将该变量对应数据转换并打印。

###### 2) 流程图



## 5.2 INPUT

### 1) 说明

输入子程序。首先将用于存储快递量数据的数组 NUM 首地址置于基址寄存器 BX，之后进行条件循环，预选输入周工作日 N 天，循环 N 次后返回 MAIN。

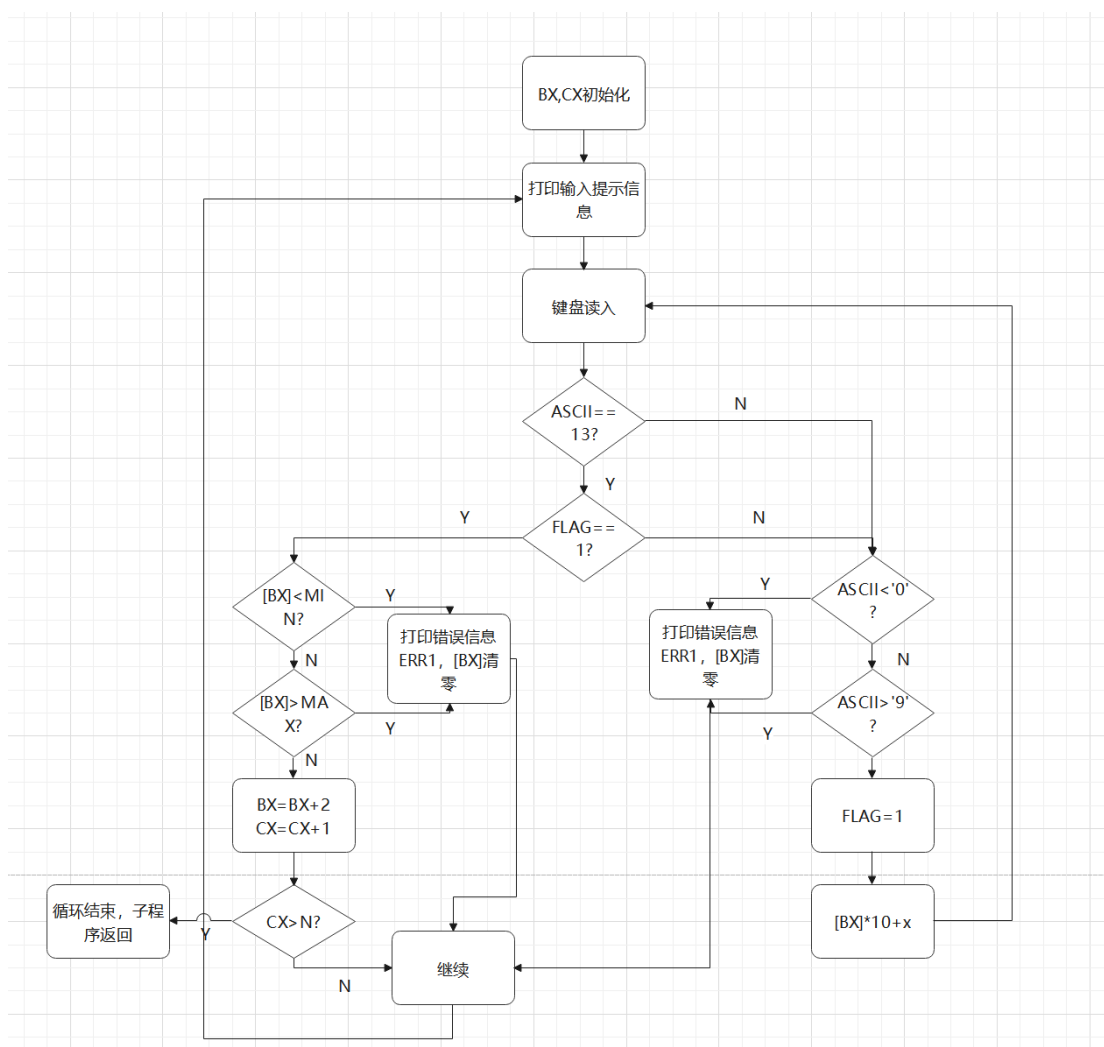
输入数据之前，首先调用 DOS 功能，输出输入提示字符串；输入快递量过程中通过 DOS 功能读取字符，直到读取到回车符，转条件判断：倘若 FLAG==1，即已读取了数字字符，结束本次读取，否则忽略。

读取到非数字字符（'0' ~ '9'）打印错误信息，并且清空数组，重新开始输入循环。

读取到合法字符即数字字符，将字符 ASCII 码转换成二进制数字，并更新数组。

当输入结束后，对数组数据进行区间范围的判断，用于剔除不合理的数据，如 999。若不在范围内则将错误信息显示在屏幕上并清零当前数组元素，之后重新进行当天的快递量输入；若在范围内则将 BX、CX 的值进行更新，便于继续输入之后的快递量。

## 2) 流程图



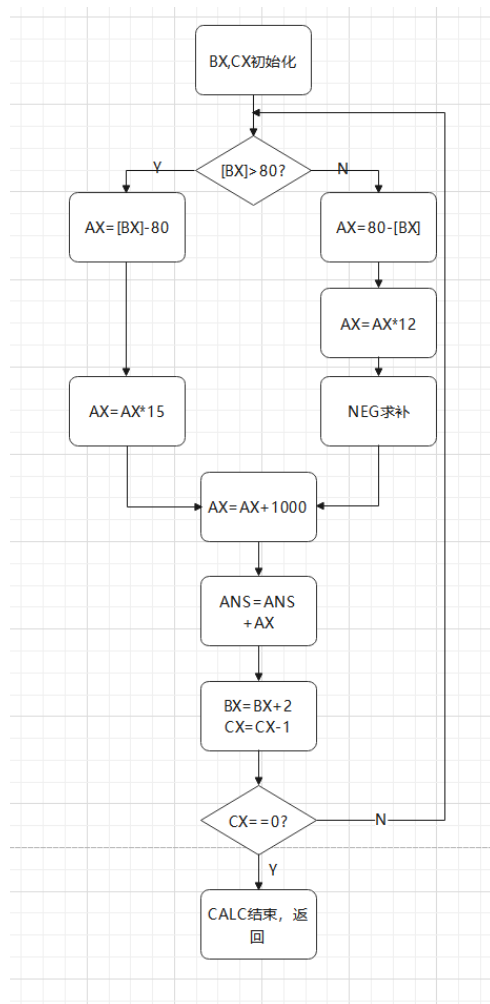
## 5.3 CALC

### 1) 说明

计算子程序。首先将 NUM 数组首地址送入基址寄存器 BX，将循环寄存器 CX 置数位工作天数，之后利用计数循环 LOOP，进行累加，当 CX 减至 0 时返回 MAIN。

对于每一次循环，需要判断当前 [BX] 与 80 的大小，若超过 80，则存在奖金，需要将该值-80 存于 AX 中，再将  $AX \times 1.5$ ；否则，用  $80 - \text{该值}$ ，结果  $\times 1.2$  存于 AX 取补（即取相反数）。累加完成后加上基本周工资 600，即可得出实际周工资。

## 2) 流程图

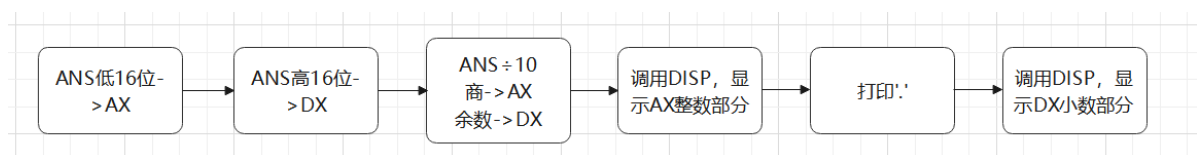


## 5.4 OUTPUT

### 1) 说明

输出子程序。调用 DOS 功能，打印输出提示的字符串，将双字变量 ANS 高 16 位送入 DX 寄存器，低 16 位送入 AX 寄存器。因为最终结果保留 1 位小数（最多也只可能为 1 为小数），因此在 CALC 子程序过程中是按照角为单位计算，扩大了 10 倍。这里应当除以 10，AX 为商，DX 为余数，前者为整数部分，后者为 1 位小数，调用两次 DISP 打印计算结果。

### 2) 流程图



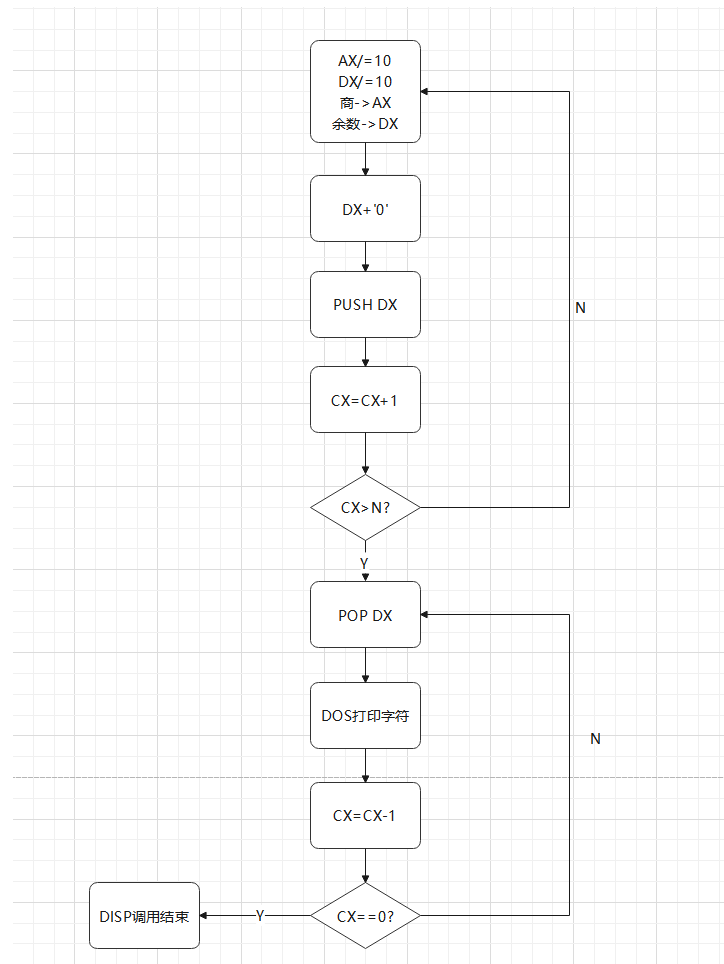
## 5.5 DISP

### 1) 说明

将内存中的数据以 10 进制方式显示，并打印在屏幕上。初始情况，数据存放于 AX 寄存器，使用条件循环，每次将 DX 和 AX 中数据  $\div 10$ ，将 DX 中数据+30H 转换为数字字符，压栈保存，实现十进制形式从低位到高位依次入栈。同时用 CX 记录压栈次数。当 AX==0 时结束循环。

根据 CX 中的压栈次数，利用 LOOP 计数循环，从栈顶依次弹出，从而实现高位到低位的顺序打印。

### 2) 流程图



### 三、调试说明

#### 1. 调试情况

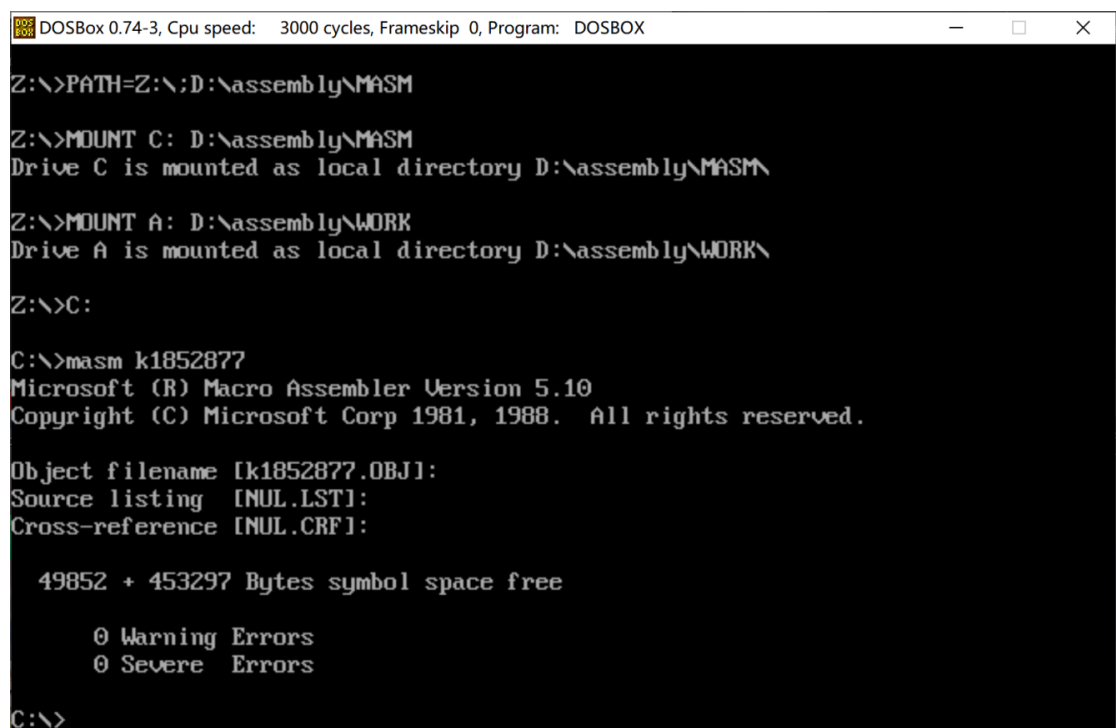
对于本程序的设计，做得较好的有：

具有较好的适用性，对于类似的问题，倘若修改工作日、基准件数、基本工资，均可以从数据段的定义进行简单修改即可适用。

DATA	SEGMENT		
N	EQU	6	;统计的天数
STA	EQU	80	;每日基准件数
BAS	EQU	1000	;每日基本工资，单位：角
MORE	EQU	15	;超出部分每件奖励工资，单位：角
LESS	EQU	12	;不足部分每件扣除工资，单位：角
MIN	EQU	0	;件数下限
MAX	EQU	999	;件数上限

同时，对程序进行了较好的子程序划分，使得程序精简易于理解、冗余代码较少。

调试运行截图：



```
DOSBox 0.74-3, Cpu speed: 3000 cycles, Frameskip 0, Program: DOSBOX

Z:\>PATH=Z:\;D:\assembly\MASM

Z:\>MOUNT C: D:\assembly\MASM
Drive C is mounted as local directory D:\assembly\MASM\

Z:\>MOUNT A: D:\assembly\WORK
Drive A is mounted as local directory D:\assembly\WORK\

Z:\>C:

C:\>masm k1852877
Microsoft (R) Macro Assembler Version 5.10
Copyright (C) Microsoft Corp 1981, 1988. All rights reserved.

Object filename [k1852877.OBJ]:
Source listing [NUL.LST]:
Cross-reference [NUL.CRF]:

49852 + 453297 Bytes symbol space free

0 Warning Errors
0 Severe Errors

C:\>
```



## 2. 连接的要求说明

本题的程序并没有采用模块化，即所有的子程序均在 K1852877.asm 一个文件中。连接截图如下：（程序没有划分堆栈段，存在该 warning）

```
C:\>link k1852877

Microsoft (R) Overlay Linker Version 3.64
Copyright (C) Microsoft Corp 1983-1988. All rights reserved.

Run File [K1852877.EXE]:
List File [NUL.MAP]:
Libraries [LIB]:
LINK : warning L4021: no stack segment
```

## 3. 数据测试

### 3.1 第一组

使用题目提供数据：102, 90, 67, 89, 98, 125。

期望结果： $22 \times 1.5 + 10 \times 1.5 - 13 \times 1.2 + 9 \times 1.5 + 18 \times 1.5 + 45 \times 1.5 + 600 = 740.4$

实际运行结果：正确

```
C:\>k1852877
Delivery Num of day 1 (0 ~ 999) : 102
Delivery Num of day 2 (0 ~ 999) : 90
Delivery Num of day 3 (0 ~ 999) : 67
Delivery Num of day 4 (0 ~ 999) : 89
Delivery Num of day 5 (0 ~ 999) : 98
Delivery Num of day 6 (0 ~ 999) : 125
The total wage is : 740.4
```

### 3.2 第二组

着重在分界线 80 附近选取数据进行测试：79, 80, 81, 82, 78, 83。

期望结果： $(1+2+3) \times 1.5 - (1+2) \times 1.2 + 600 = 605.4$

实际运行结果：正确

```
C:\>k1852877
Delivery Num of day 1 (0 ~ 999) : 79
Delivery Num of day 2 (0 ~ 999) : 80
Delivery Num of day 3 (0 ~ 999) : 81
Delivery Num of day 4 (0 ~ 999) : 82
Delivery Num of day 5 (0 ~ 999) : 78
Delivery Num of day 6 (0 ~ 999) : 83
The total wage is : 605.4
```

### 3.3 第三组

考虑极大数据 999。

期望结果=  $(999-80) * 1.5 * 6 + 600 = 8871.0$

实际运行结果：正确

```
C:\>k1852877
Delivery Num of day 1 (0 ~ 999) : 999
Delivery Num of day 2 (0 ~ 999) : 999
Delivery Num of day 3 (0 ~ 999) : 999
Delivery Num of day 4 (0 ~ 999) : 999
Delivery Num of day 5 (0 ~ 999) : 999
Delivery Num of day 6 (0 ~ 999) : 999
The total wage is : 8871.0
```

### 3.4 第四组

考虑极小数据 0。

期望结果=  $600 - 6 * 1.2 * 80 = 24.0$

实际运行结果：正确

```
Delivery Num of day 1 (0 ~ 999) : 0
Delivery Num of day 2 (0 ~ 999) : 0
Delivery Num of day 3 (0 ~ 999) : 0
Delivery Num of day 4 (0 ~ 999) : 0
Delivery Num of day 5 (0 ~ 999) : 0
Delivery Num of day 6 (0 ~ 999) : 0
The total wage is : 24.0
```

### 3.5 第五组

错误数据输入，包括非数字字符以及超出上下限内容：1000，? 空格，回车。

实际运行结果：正确。并且超限的报错信息与非法字符的报错信息有区分。

且在没有任何数字的情况下输入回车，程序继续等待数字字符的键入。

```
C:\>k1852877
Delivery Num of day 1 (0 ~ 999) : 1000
Numbers out of range, please retry.
Delivery Num of day 1 (0 ~ 999) : ?
Illegal character contained, please retry.
Delivery Num of day 1 (0 ~ 999) : 
Illegal character contained, please retry.
Delivery Num of day 1 (0 ~ 999) :
```

## 4. 运行结果分析

本程序对一般数据、临界数据均能给出正确的运算结果。并且能够判断非法字符和超出范围两种数据错误，并能够给出对应的提示信息。

## 四、使用说明

这是程序提供给用户使用，必须作出的说明。如：

- 1.程序运行的软硬件环境、适用范围。
- 2.程序的使用方法、调试方法、操作步骤等。
- 3.要求输入信息的类型及格式。
- 4.出错信息的含义及注意事项等。

### 1. 运行环境

操作系统：Windows10 64 位

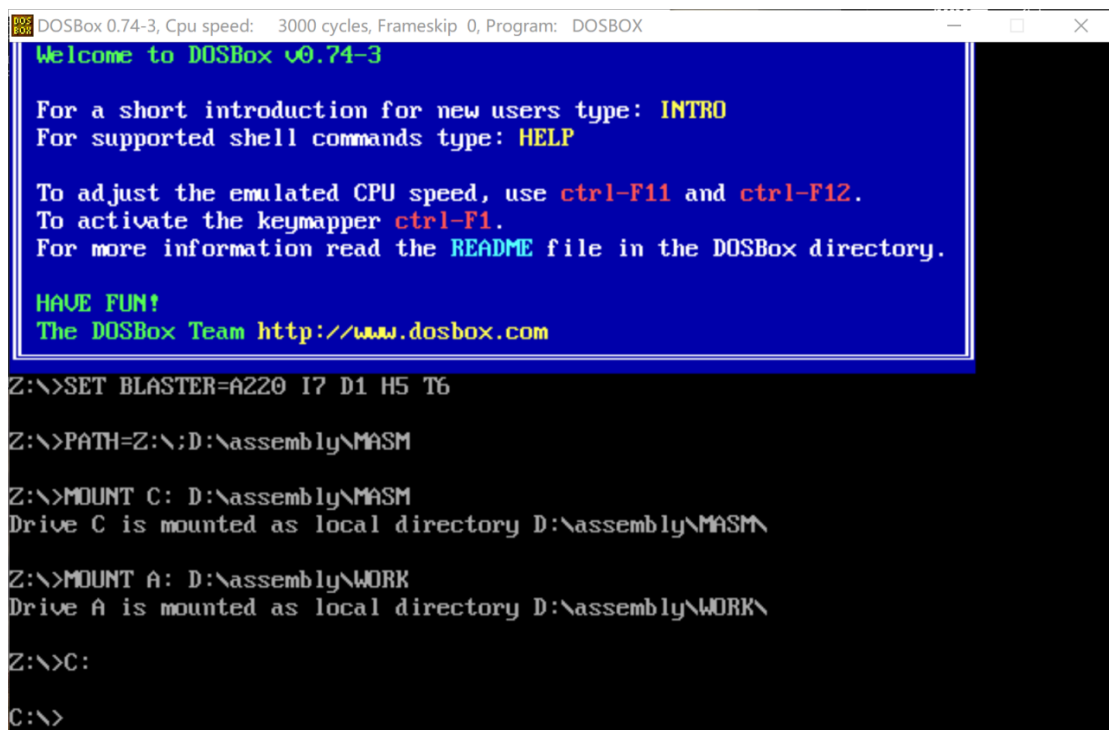
汇编软件：DOSBox0.74

硬件环境：AMD Ryzen 7 4800H

### 2. 程序使用

1) 将 K1852877.asm 文件，放置于 MASM 目录下

2) 启动 DOSBox 输入指令如下图：（路径替换为实际工作路径）



The screenshot shows a DOSBox 0.74-3 window. The title bar reads "DOSBox 0.74-3, Cpu speed: 3000 cycles, Frameskip 0, Program: DOSBOX". The main window has a blue background with white text. The text reads: "Welcome to DOSBox v0.74-3", "For a short introduction for new users type: INTRO", "For supported shell commands type: HELP", "To adjust the emulated CPU speed, use ctrl-F11 and ctrl-F12.", "To activate the keymapper ctrl-F1.", "For more information read the README file in the DOSBox directory.", "HAVE FUN!", "The DOSBox Team http://www.dosbox.com". Below this, the command prompt shows the following commands and their outputs: "Z:\>SET BLASTER=A220 I7 D1 H5 T6", "Z:\>PATH=Z:\;D:\assembly\MASM", "Z:\>MOUNT C: D:\assembly\MASM", "Drive C is mounted as local directory D:\assembly\MASM\", "Z:\>MOUNT A: D:\assembly\WORK", "Drive A is mounted as local directory D:\assembly\WORK\", "Z:\>C:", "C:\>".

3) 输入命令 MASM K1852877.asm, 进行汇编, 并回车。

```
C:\>MASM K1852877.asm
Microsoft (R) Macro Assembler Version 5.10
Copyright (C) Microsoft Corp 1981, 1988. All rights reserved.

Object filename [K1852877.OBJ]:
Source listing [NUL.LST]:
Cross-reference [NUL.CRF]:

    49852 + 455345 Bytes symbol space free

    0 Warning Errors
    0 Severe Errors

C:\>a
```

4) 输入命令 LINK K1852877.obj, 进行连接, 并回车。

```
C:\>LINK K1852877.obj

Microsoft (R) Overlay Linker Version 3.64
Copyright (C) Microsoft Corp 1983-1988. All rights reserved.

Run File [K1852877.EXE]:
List File [NUL.MAP]:
Libraries [LIB]:
LINK : warning L4021: no stack segment

C:\>_
```

5) 运行程序, 输入命令 K1852877.exe 即可。

```
C:\>K1852877.exe
Delivery Num of day 1 (0 ~ 999) :
```

### 3. 输入信息格式

- 1) 需要输入十进制的无符号非负整数, 可以有前缀 0, 如“085”, 但不能写为“+85”。
- 2) 键入的数据范围在 0~999 之间, 超出范围将会有“out of range”的报错提示。
- 3) 键入的数据不能包括非数字字符, 否则将会有“illegal character”的报错提示。
- 4) 每键入一个数据后, 用回车表示结束该数据的输入。

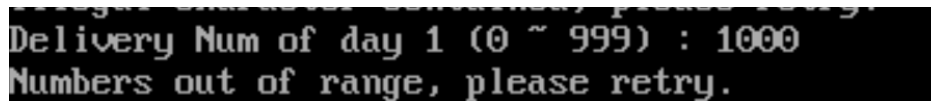
### 4. 错误信息含义与注意事项

#### 1) 非法字符

```
Delivery Num of day 1 (0 ~ 999) : c
illegal character contained, please retry.
```

表示输入了非‘0’~‘9’的数字字符, 需要重新输入当天快递量。

## 2) 超出范围



表示输入数据为合法十进制数字，但是超出了 0-999 的范围，需要重新输入当天快递量。

# 五、课程总结

## 1. 心得体会

在连续两周的“汇编语言程序设计”课程中，毫无疑问，知识的密度相当之高。要在短短两周掌握汇编语言的大概，其实还是具有相当的挑战性。特别是课程的安排，正好是我刚刚结束最后一门答辩的三天后，心情仍然是浮躁的，而且汇编课程一上就是一整个上午，也令我在前几天的课程学习中并没有很好的进入状态，这是我需要检讨自己的地方。

由于转专业的缘故，需要补修电信学院绝大多数同学在大一暑假上的课，在此之前，我已经学习过计算机组成原理，用 Verilog 语言写过 MIPS54 CPU，可以说对于汇编指令、计算机组成相关内容掌握得较快。尽管在前一两次上机课还没有找回良好的学习状态，但是张光长老师真的非常负责任，经常手把手地教刚入门的同学写代码，同样地也大大帮助了我，令我非常感动。

因此，在课程学习的中后段，我均保持了全勤以及上机作业、纸质作业均按时提交，可以说能够独立地结局很多问题。

不得不说，张光长老师的教学方式令我感觉非常受用，张老师并没有局限于 PPT 和“死知识”，而是经常性地理论课带同学进行代码编写，并亲自演示和 debug，课后，也会在群里分享一些其他课外的习题供我们练习，并抽时间讲解而非弃之不顾。哪怕是很晚的群聊讨论，张老师也时时刻刻保持关注，给出令人满意的答复。因此，张老师负责的态度，和重视实践的教学方式令我的基础知识学习较为扎实，动手能力也不弱。因此课程的上机题，我均为独立完成。

至于参考资料，我之前有接触过清华大学的《计算机组成原理》，在课程学习阶段，我主要参考了《汇编语言程序设计》这本书的相关章节以更加详尽地了解 PPT 上的内容。对于纸质作业，一般来说难度较低，基本无需参考资料，通过上课听讲即可做出，而准备理论考试的阶段，我主要针对 PPT 的内容进行了着重的复习以及抄录，从而加深印象。

不得不说，本节课我的收获还是远远大于预期，本以为已经写过 CPU，对汇编指令有一定了解，实际上本次课程不仅令我更深地理解了计算机的结构，而且还了解了很多编程技巧。我对数据在内存中的存储（尤其是无符号数和有符号数其实在内存中并不加区分）、小端序排列、代码段数据段、堆栈的使用都有了更深的理解。对于计算机如何处理溢出、扩充以及数据传输和硬件结构的关系也有了新的认识。不过不足之处仍有很多，对于比较复杂的传参、现场保护、恢复还是很容易出错，对于分支程序设计的结构还存在着精简优化的空间。而且对于输入输出指令并没有进行编程实践，而是使用 DOS 功能实现的。这也是我的不足之

处。

不过，这门课程总体而言还是很有必要的，汇编语言是高级语言和硬件机器语言之间的桥梁，老师教学也是选取了核心内容讲解，虽然内容很多，但也算得上轻重分明、详细易懂。

在课程结束后一周之内，我就完成了大作业，很遗憾并没有留下 DEBUG 时的截图供总结，不过也让我深深意识到，处理越是底层的東西，就应该考虑越多，尤其是汇编语言中对内存的管理，稍有不慎就会出现错误，同时脱离了 VS2019 这样有利的 DEBUG 工具，对于编程者的细致程度都有了更高的要求。

在 DEBUG 时，我参考了一些 CSDN 的内容，出于兴趣，我了解了将常量定义 EQU 实际上有很大的优势：因为如果使用 DB, DW 等类型进行变量定义，在使用的过程中，仍然涉及存储器的传输，而 EQU 则相当于简单的替换，这也是一个提升运行速度的小技巧。

不得不说，在学习了 C 语言和编写了 MIPS CPU 后重新学习当初草草补习的汇编语言，还是令我收获很多，结合计组的硬件知识，能让我明白例如为什么只能有一个数存在于主存储器（通路占用），结合 C 语言的知识，又能让我很好地理解循环结构与分支结构，同时，摆脱了编译器报错，也令我开始培养自己慎重、仔细的编码态度，这令我收获颇丰。

最后，感谢张光长老师对我的悉心指导，让我相对顺利地入门汇编程序设计，有了一点自己的浅见拙识，并融汇、加深了对硬件课、C 语言的认识，感谢您的付出与关怀！

## 六、程序清单

K1852877.asm - 记事本

文件(F) 编辑(E) 格式(O) 查看(V) 帮助(H)

;1852877 赵昊堃 计时工资计算

```
DATA    SEGMENT
N        EQU        6            ;统计的天数
STA      EQU        80          ;每日基准件数
BAS      EQU        1000        ;每日基本工资，单位：角
MORE     EQU        15          ;超出部分每件奖励工资，单位：角
LESS     EQU        12          ;不足部分每件扣除工资，单位：角
MIN      EQU        0           ;件数下限
MAX      EQU        999         ;件数上限
C10      DW        10
NUM      DW        N DUP(0)
ANS      DD        0
FLAG     DB          0          ;标志变量，记录是否读到过回车外的字符
PROMPT1  DB        "Delivery Num of day ", '$'
PROMPT2  DB        " (", '$'
PROMPT3  DB        " ~ ", '$'
PROMPT4  DB        ") : ", '$'
RESULT   DB        "The total wage is : ", '$'
ERRMSG1  DB        0AH,0DH,"illegal character contained, please retry.",0AH,0DH,'$'
ERRMSG2  DB        "Numbers out of range, please retry.",0AH,0DH,'$'
DATA     ENDS
CODE     SEGMENT
        ASSUME     CS:CODE,DS:DATA
```

```
;-----
;子程序名: DISP
;功能: 将寄存器AX中的数据以十进制显示出来
;入口参数: AX
;出口参数: 无
;-----
DISP     PROC        FAR
        PUSH        DX
        PUSH        CX
        PUSH        BX
        MOV         CX,0      ;计数器
        MOV         BX,10
REP1:    MOV         DX,0
        DIV         BX        ;除以10取余
        ADD         DX,30H     ;DX加30H
        PUSH        DX        ;先低后高入栈
        INC         CX        ;计余数个数
        OR          AX,AX
        JNZ         REP1      ;商0结束循环
REP2:    POP         DX        ;先高后低弹出
        MOV         AH,2      ;显示
        INT         21H
        LOOP        REP2
        POP         BX
        POP         CX
        POP         DX
        RET
DISP     ENDP
```

```

;-----
;子程序名: INPUT
;功能: 从键盘输入一周中每天的快递量并进行错误处理
;入口参数: 无
;出口参数: NUM
;-----
INPUT    PROC    FAR
        LEA     BX,NUM
        MOV     CX,1
L1:      LEA     DX,PROMPT1
        MOV     AH,9
        INT     21H
        MOV     AX,CX
        CALL    DISP
        LEA     DX,PROMPT2
        MOV     AH,9
        INT     21H
        MOV     AX,MIN
        CALL    DISP
        LEA     DX,PROMPT3
        MOV     AH,9
        INT     21H
        MOV     AX,MAX
        CALL    DISP
        LEA     DX,PROMPT4
        MOV     AH,9

L2:      INT     21H
        MOV     AH,1
        INT     21H
        CMP     AL,0DH
        JE      DONE
        CMP     AL,39H
        JA      ERROR1
        CMP     AL,30H
        JB      ERROR1
        MOV     FLAG,1
        MOV     DX,AX      ;输入字符转移到CX寄存器
        AND     DX,000FH;转换成二进制数
        MOV     AX,[BX]
        PUSH    DX
        MUL     C10
        POP     DX
        ADD     AX,DX      ;新输入数字拼接到已输入数字中
        MOV     [BX],AX
        JMP     L2
DONE:    CMP     FLAG,0
        JE      L2
        CMP     WORD PTR [BX],MAX
        JA      ERROR2
        CMP     WORD PTR [BX],MIN
        JB      ERROR2
        INC     BX
        INC     BX
        INC     CX
        CMP     CX,N

```



```

        JA      EXIT
        MOV     FLAG,0
        JMP     L1
ERROR1: LEA     DX,ERRMSG1
        MOV     AH,9
        INT     21H
        MOV     WORD PTR [BX],0
        MOV     FLAG,0
        JMP     L1
ERROR2: LEA     DX,ERRMSG2
        MOV     AH,9
        INT     21H
        MOV     WORD PTR [BX],0
        MOV     FLAG,0
        JMP     L1
EXIT:    RET
INPUT   ENDP

```

```

;-----
;子程序名: CALC
;功能: 根据各日件数计算总工资
;入口参数: BX
;出口参数: ANS
;-----

```

```

CALC    PROC
        LEA     BX,NUM
        MOV     CX,N
L3:      MOV     AX,[BX]
        CMP     AX,STA
        JA      EL
        MOV     AX,STA
        SUB     AX,[BX]
        MOV     DX,LESS
        MUL     DX
        NEG     AX
        JMP     AC
EL:      SUB     AX,STA
        MOV     DX,MORE
        MUL     DX
AC:      ADD     AX,BAS
        ADD     WORD PTR [ANS],AX
        ADC     WORD PTR [ANS+2],0
        INC     BX
        INC     BX
        LOOP    L3
        RET
CALC    ENDP

```

```

;-----
;子程序名: OUTPUT
;功能: 输出总工资, 保留一位小数
;入口参数: ANS
;出口参数: 无
;-----

```

```

OUTPUT  PROC
        LEA     DX,RESULT
        MOV     AH,9
        INT     21H
        MOV     AX,WORD PTR [ANS]
        MOV     DX,WORD PTR [ANS+2]

```

```

        DIV     C10
        PUSH    DX
        CALL    DISP
        MOV     AH,2
        MOV     DL,','
        INT     21H
        POP     AX
        CALL    DISP
        RET
OUTPUT  ENDP
;-----
;子程序名: MAIN
;功能: 主程序
;入口参数: 无
;出口参数: 无
;-----
MAIN    PROC    FAR
        PUSH    DS
        MOV     AX,0
        PUSH    AX
        MOV     AX,DATA
        MOV     DS,AX
        CALL    INPUT
        CALL    CALC
        CALL    OUTPUT
        RET
MAIN    ENDP
CODE    ENDS
        END     MAIN
```