1. **МИНОБРНАУКИ РОССИИ**
2. **Федеральное государственное бюджетное**
3. **образовательное учреждение высшего образования**
4. **«Тульский государственный университет»**
5. Институт прикладной математики и компьютерных наук
6. Кафедра «Вычислительная техника»

Отчет по лабораторной работе №2

## «РАСПРЕДЕЛЕНИЕ ПАМЯТИ. КОМПОНОВКА ПРОГРАММ НА ЯЗЫКЕ АССЕМБЛЕРА»

По дисциплине «Операционные системы и среды»

Выполнили студенты группы 220681:  
Илюхин Д.М.

Яковлев С.С.  
Проверил:  
Доц. Андриянова М. А.

1. Тула 2020

# ЦЕЛЬ РАБОТЫ

Целью работы является раскрытие технологии программирования, включающей компоновку и выполнение ассемблерных программ. Задачами работы является написание внешних подпрограмм на Ассемблере и компоновка их в один исполняемый модуль.

# ЗАДАНИЕ

Вычислить разность двух чисел, оформив это в виде основной программы и внешней подпрограммы тремя способами.

Для каждого способа выполнить следующие действия:

1. Оттранслировать оба модуля.

2. Просмотреть листинги трансляции, обратить внимание на таблицы идентификаторов и карты компоновки.

3. Создать выполняемый модуль.

4. При помощи отладчика выполнить программу. Определить действительный адрес кодового сегмента для подпрограммы (в команде CALL).

**3. ХОД РАБОТЫ**

В начале необходимо создать текстовый файл для основной программы и открыть его. После в нём нужно объявить внешнюю подпрограмму SubDif с параметром far, сегмент стека, сегмент данных с сообщениями и переменными значений (n1, n2) и сегмент кода. В сегменте кода нужно подключить все ранее объявленные сегменты и переместить адрес сегмента данных в регистр ds. После выводится сообщение, и пользователь должен ввести строку, которая разбивается на отдельные символы и получает из них число, которое заносится в переменную n1. Если преобразование прошло безуспешно, то вызывается участок кода, который выводит сообщение об ошибку и производит выход из программы. Таким же образом получаем второе число и заносим его в n2. После значения переменных переносятся в регистры ax, bx и вызывается подпрограмма SubDif. После производится выход из программы и закрываются ранее открытый сегмент.

Теперь надо создать второй текстовый файл для внешней подпрограммы и открыть его. В нём прописываем сегмент данных (сообщение и переменные n1, n2, r) и сегмент кода. Подключаем сегменты и открываем доступ к подпрограмме командой public. Переносим значения из регистров ax и bx в n1 и n2. После проверяем какое число больше и переходим к участку кода, где из большего вычитаю меньшее. После выводим сообщение и результат посимвольно. После осуществляется выход из программы и закрытие сегмента кода.

Далее оттранслируем файлы, создав при этом листинг трансляции для дальнейшего изучения. При компоновке укажем /map, чтобы получить карту компоновки. Таблица идентификаторов для первого варианта представлена на рисунке 1. В другой программе таблица отличается лишь переменными и значениями.

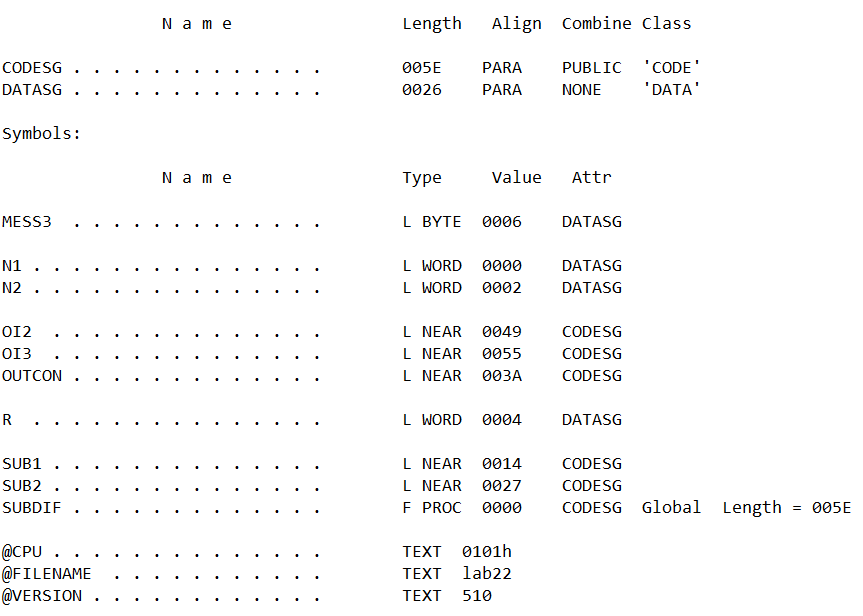


Рисунок 1 – Таблица идентификаторов в первом варианте

В карте компоновки (Рисунок 2) представлена организация программы в памяти. Сегменты кода и данных из разных программ разделены и имеют разные значения.

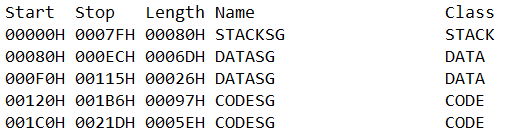


Рисунок 2 – Карта компоновки первого варианта

Теперь изменим программы, написав в каждой из них public в объявлении сегментов кода, тем самым объединив их на этапе компоновки. Это можно увидеть в карте компоновки (Рисунок 3), так как вместо двух сегментов кода остался один.

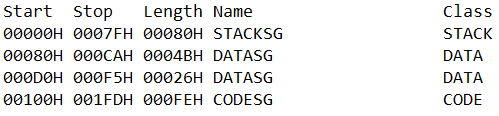


Рисунок 3 – Карта компоновки второго варианта

В листинге программ таблицы идентификаторов не изменились.

Теперь ещё раз изменим программы, написав в каждой из них public в объявлении сегментов данных и прописав общие переменные, тем самым объединив их на этапе компоновки. Из-за этого можно убрать перемещение данных из переменных в регистры и обратно. Об объединении сегментов данных можно судить по карте компоновки (Рисунок 4), где вместо двух сегментов данных остался один.

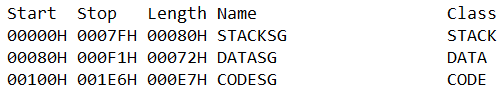


Рисунок 4 – Карта компоновки третьего варианта

Также в таблице идентификаторов основной программы (Рисунок 5) определённые переменные помечены как глобальные.



Рисунок 5 – Таблица идентификаторов в основной программе

В таблице идентификаторов внешней подпрограммы (Рисунок 6) определённые переменные помечены как внешние.



Рисунок 6 – Таблица идентификаторов внешней подпрограммы

# 4. ТЕСТИРОВАНИЕ

При запуске программы и вводе чисел 123 и 56 в консоль выводится следующее сообщение (Рисунок 7).

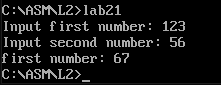


Рисунок 7 – Результат программы

# 5. ВЫВОД

В ходе работы была изучена компоновка и выполнение ассемблерных программ, написание внешних подпрограмм на Ассемблере и компоновка их в один исполняемый модуль.

**Приложение**

**lab21.asm**

Title DIF (EXE)

Extrn SubDif:Far

StackSg Segment PARA STACK 'Stack'

DW 64 DUP(?)

StackSg ENDS

DataSg Segment PARA 'Data'

mess1 DB 'Input first number: ','$'

mess2 DB 'Input second number: ','$'

buff db 6,7 Dup(?)

error DB 'Error, all are bad ','$'

n1 dw 1

n2 dw 1

DataSg ENDS

CodeSg Segment PARA 'Code'

Begin Proc FAR

Assume CS:CodeSg, DS:DataSg, SS:StackSg

Push DS

Sub AX, AX

Push AX

Mov AX, DataSg

Mov DS, AX

mov ah,9

mov dx,OFFSET mess1

int 21h

input1:

mov ah,0ah

xor di,di

mov dx,offset buff

int 21h

mov dl,0ah

mov ah,02

int 21h

mov si,offset buff+2

xor ax,ax

mov bx,10

ii1:

mov cl,[si]

cmp cl,0dh

jz endin1

cmp cl,'0'

jb er

cmp cl,'9'

ja er

sub cl,'0'

mul bx

add ax,cx

inc si

jmp ii1

endin1:

mov n1, ax

mov ah,9

mov dx,OFFSET mess2

int 21h

input2:

mov ah,0ah

xor di,di

mov dx,offset buff

int 21h

mov dl,0ah

mov ah,02

int 21h

mov si,offset buff+2

xor ax,ax

mov bx,10

ii2:

mov cl,[si]

cmp cl,0dh

jz endin2

; если символ не последний, то проверяем его на правильность

cmp cl,'0'

jb er

cmp cl,'9'

ja er

sub cl,'0'

mul bx

add ax,cx

inc si

jmp ii2

er:

mov dx, offset error

mov ah,09

int 21h

int 20h

endin2:

mov n2, ax

mov bx, n2

mov ax, n1

Call SubDif

Ret

Begin ENDP

CodeSg ENDS

END Begin

**lab21.asm**

Title SubDif

DataSg Segment PARA 'Data'

n1 dw 1

n2 dw 1

r dw 1

mess3 DB 10,13,'The difference of two number ','$'

DataSg ENDS

CodeSg Segment PARA 'Code'

SubDif Proc FAR

Assume CS:CodeSg, DS:DataSg

Public SubDif

mov n2, bx

mov n1, ax

mov bx, n2

cmp n1, bx

ja sub1

jmp sub2

sub1:

mov bx, n2

sub n1, bx

mov bx, n1

mov r, bx

jmp outcon

sub2:

mov bx, n1

sub n2, bx

mov bx, n2

mov r, bx

jmp outcon

outcon:

mov ah,9

mov dx,OFFSET mess3

int 21h

mov ax, r

xor cx, cx

mov bx, 10

oi2:

xor dx,dx

div bx

push dx

inc cx

test ax, ax

jnz oi2

mov ah, 02h

oi3:

pop dx

add dl, '0'

int 21h

loop oi3

Ret

SubDif ENDP

CodeSg ENDS

END SubDif