

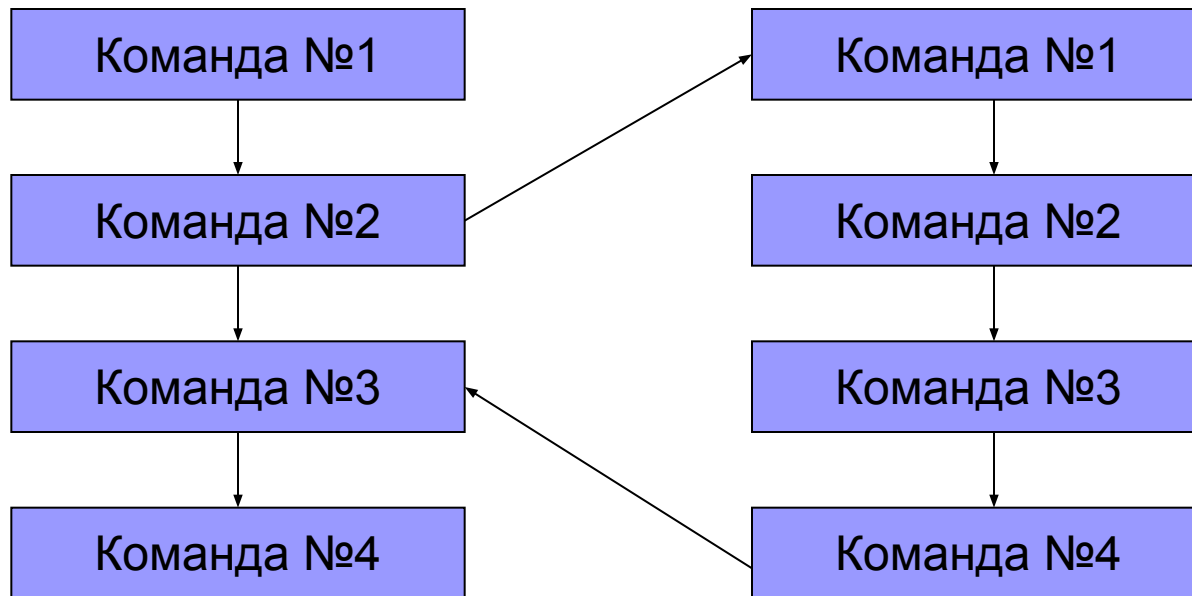


# Подпрограммы

# Выполнение подпрограмм

Основная программа

Подпрограмма



# Команды процессора для работы с подпрограммами

- Вызов подпрограммы – **CALL**
- Возврат из подпрограммы – **RET**

*команды относятся к командам передачи управления*

# Команды процессора для работы с подпрограммами

Программа

|     |          |
|-----|----------|
| 012 | MOV ...  |
| 013 | CALL 567 |
| 014 | MOV ...  |

# Команды процессора для работы с подпрограммами

Программа

|     |          |
|-----|----------|
| 012 | MOV ...  |
| 013 | CALL 567 |
| 014 | MOV ...  |

|     |     |
|-----|-----|
| EIP | 012 |
|-----|-----|

# Команды процессора для работы с подпрограммами

Программа

|     |          |
|-----|----------|
| 012 | MOV ...  |
| 013 | CALL 567 |
| 014 | MOV ...  |

|     |     |
|-----|-----|
| EIP | 013 |
|-----|-----|

# Команды процессора для работы с подпрограммами

Программа

|     |          |
|-----|----------|
| 012 | MOV ...  |
| 013 | CALL 567 |
| 014 | MOV ...  |

|     |     |
|-----|-----|
| EIP | 014 |
|-----|-----|

# Команды процессора для работы с подпрограммами

Программа

|     |                 |
|-----|-----------------|
| 012 | MOV ...         |
| 013 | <b>CALL 567</b> |
| 014 | MOV ...         |

|     |     |
|-----|-----|
| EIP | 014 |
|-----|-----|

Подпрограмма

|     |         |
|-----|---------|
| 567 | MOV ... |
| 568 | ADD ... |
| 569 | RET     |



# Команды процессора для работы с подпрограммами

Программа

|     |                 |
|-----|-----------------|
| 012 | MOV ...         |
| 013 | <b>CALL 567</b> |
| 014 | MOV ...         |

|     |     |
|-----|-----|
| EIP | 014 |
|-----|-----|

Подпрограмма

|     |         |
|-----|---------|
| 567 | MOV ... |
| 568 | ADD ... |
| 569 | RET     |

Стек

|     |
|-----|
| 014 |
|-----|

# Команды процессора для работы с подпрограммами

Программа

|     |                 |
|-----|-----------------|
| 012 | MOV ...         |
| 013 | <b>CALL 567</b> |
| 014 | MOV ...         |

|     |     |
|-----|-----|
| EIP | 567 |
|-----|-----|

Подпрограмма

|     |         |
|-----|---------|
| 567 | MOV ... |
| 568 | ADD ... |
| 569 | RET     |

Стек

|     |
|-----|
| 014 |
|-----|

# Команды процессора для работы с подпрограммами

Программа

|     |          |
|-----|----------|
| 012 | MOV ...  |
| 013 | CALL 567 |
| 014 | MOV ...  |

|     |     |
|-----|-----|
| EIP | 568 |
|-----|-----|

Подпрограмма

|     |         |
|-----|---------|
| 567 | MOV ... |
| 568 | ADD ... |
| 569 | RET     |

Стек

|     |
|-----|
| 014 |
|-----|

# Команды процессора для работы с подпрограммами

Программа

|     |          |
|-----|----------|
| 012 | MOV ...  |
| 013 | CALL 567 |
| 014 | MOV ...  |

|     |     |
|-----|-----|
| EIP | 569 |
|-----|-----|

Подпрограмма

|     |                |
|-----|----------------|
| 567 | MOV ...        |
| 568 | <b>ADD ...</b> |
| 569 | RET            |

Стек

|     |
|-----|
| 014 |
|-----|

# Команды процессора для работы с подпрограммами

Программа

|     |          |
|-----|----------|
| 012 | MOV ...  |
| 013 | CALL 567 |
| 014 | MOV ...  |

|     |     |
|-----|-----|
| EIP | 570 |
|-----|-----|

Подпрограмма

|     |         |
|-----|---------|
| 567 | MOV ... |
| 568 | ADD ... |
| 569 | RET     |

Стек

|     |
|-----|
| 014 |
|-----|

# Команды процессора для работы с подпрограммами

Программа

|     |          |
|-----|----------|
| 012 | MOV ...  |
| 013 | CALL 567 |
| 014 | MOV ...  |

|     |     |
|-----|-----|
| EIP | 014 |
|-----|-----|

Подпрограмма

|     |         |
|-----|---------|
| 567 | MOV ... |
| 568 | ADD ... |
| 569 | RET     |

# Команды процессора для работы с подпрограммами

Программа

|     |          |
|-----|----------|
| 012 | MOV ...  |
| 013 | CALL 567 |
| 014 | MOV ...  |

|     |     |
|-----|-----|
| EIP | 015 |
|-----|-----|

Подпрограмма

|     |         |
|-----|---------|
| 567 | MOV ... |
| 568 | ADD ... |
| 569 | RET     |

# Где описываются программы

**MOV ...**

**Procedure:**

**ADD ...**

**NEG ...**

**RET**

**SUB ...**

**MULL ...**

**CALL Procedure**

**CDQ**

**IDIV ...**



# Где описываются программы

**MOV ...**

**JMP MainNext**

**Procedure:**

**ADD ...**

**NEG ...**

**RET**

**MainNext:**

**SUB ...**

**MULL ...**

**CALL Procedure**

**CDQ**

**IDIV ...**



# Где описываются программы

- Перед основной программой
- После основной программы
- В отдельном модуле

# Где описываются программы

- Перед основной программой

**.code**

**Procedure:**

; код подпрограммы

**RET**

**Start:**

; код основной программы

**CALL Procedure**

; код основной программы

**end Start**

# Где описываются программы

## ■ После основной программы

**.code**

**Start:**

**; код основной программы**

**CALL Procedure**

**; код основной программы**

**CALL ExitProcess**

**Procedure:**

**; код подпрограммы**

**RET**

**end Start**



# Способы передачи параметров в подпрограммы

- Через регистры общего назначения
- Через общую память
- Через стек

# Передача параметров через регистры общего назначения

**Factorial:**

**MOV ECX, EAX**

**MOV EBX, 2**

**MOV EAX, 1**

**Cycle:**

**CMP EBX, ECX**

**JG EndFunction**

**MUL EBX**

**INC EBX**

**JMP Cycle**

**EndFunction:**

**RET**

# Передача параметров через регистры общего назначения

**.data**

n **dd** 5

**.data?**

result **dd** ?

**.code**

**Factorial:**

; тело функции

**RET**

**Start:**

**MOV** EAX, n

**CALL** Factorial

**MOV** result, EAX

# Передача параметров через регистры общего назначения

## **Достоинства**

- легко использовать
- большая скорость работы
- можно возвращать несколько значений



# Передача параметров через регистры общего назначения

## Недостатки

- малое количество параметров
- трудности использования регистров в подпрограмме
- трудности вложенных и рекурсивных ВЫЗОВОВ

# Передача параметров через общую память

**Factorial:**

**MOV ECX, param**

**MOV EBX, 2**

**MOV EAX, 1**

**Cycle:**

**CMP EBX, ECX**

**JG EndFunction**

**MUL EBX**

**INC EBX**

**JMP Cycle**

**EndFunction: MOV param+4, EAX**

**RET**

# Передача параметров через общую память

**.data**

n **dd** 5

**.data?**

result **dd** ?

param **dd** 2 **dup**(?)

**.code**

**MOV** EAX, n

**MOV** param, EAX

**CALL** Factorial

**MOV** EAX, param+4

**MOV** result, EAX

# Передача параметров через общую память

## **Достоинства**

- произвольное количество параметров
- переменное количество параметров
- регистры общего назначения свободны

# Передача параметров через общую память

## Недостатки

- низкое быстродействие
- трудности рекурсивных вызовов

# Передача параметров через стек

## Команды работы со стеком

- **push** – помещает в вершину стека некоторое значение
- **pop** – извлекает из вершины стека некоторое значение

# Передача параметров через стек

## Команды работы со стеком

- **pusha** – помещает в вершину стека значения всех регистров общего назначения
- **popa** – извлекает из вершины стека значения всех регистров общего назначения

# Передача параметров через стек

## Команды работы со стеком

- **pushf** – помещает в вершину стека значение регистра флагов
- **popf** – извлекает из вершины стека значения регистра флагов



# Передача параметров через стек

## Стек:

- Позволяет обрабатывать только 32-разрядные числа
- Адрес вершины стека храниться в регистре ESP (смещение относительно сегментного регистра SS)
- Вершина стека, это ячейка памяти, содержащая последнее помещённое в стек значение

# Передача параметров через стек

Алгоритм работы команды **push**:

1. **add ESP, 4**
2. **mov [ESP], <источник>**

# Передача параметров через стек

Алгоритм работы команды **pop**:

1. **mov** <приёмник>, [ESP]
2. **sub** ESP, 4

# Передача параметров через стек

Алгоритм работы команды **call**:

1. **add EIP, <размер команды call>**
2. **add ESP, 4**
3. **mov [ESP], EIP**
4. **mov EIP, <адрес метки>**

# Передача параметров через стек

Алгоритм работы команды **ret**:

1. **mov EIP, [ESP]**
2. **sub ESP, 4**

# Передача параметров через стек

**Factorial:**

**MOV ECX, [ESP+8]**

**MOV EBX, 2**

**MOV EAX, 1**

**Cycle:**

**CMP EBX, ECX**

**JG EndFunction**

**MUL EBX**

**INC EBX**

**JMP Cycle**

**EndFunction: MOV [ESP+4], EAX**

**RET**

# Передача параметров через стек

**.data**

n **dd** 5

**.data?**

result **dd** ?

**.code**

**PUSH** n

**PUSH** 0

**CALL** Factorial

**POP** result

**ADD** ESP, 4



# Передача параметров через стек

## Достоинства

- произвольное количество параметров
- переменное количество параметров
- простота использования
- легкая организация рекурсии



# Передача параметров через стек

## Недостатки

- трудно отслеживать состояние стека
- после вызова подпрограммы основная программа должна ***выравнивать*** стек

# Подпрограммы в С

- параметры передаются через стек (при этом в стек параметры помещаются с конца)
- Результат возвращается в регистрах
  - 1 байт – AL
  - 2 байта – AX
  - 4 байта – EAX
  - 8 байт – (EAX, EDX)

# Функции Windows API

- после вызова функций с фиксированным числом параметров не нужно выравнивать стек

# Передача параметров с использованием STDCALL

**Factorial:**

```
MOV ECX, [ESP+4]
```

```
MOV EBX, 2
```

```
MOV EAX, 1
```

**Cycle:**

```
CMP EBX, ECX
```

```
JG EndFunction
```

```
MUL EBX
```

```
INC EBX
```

```
JMP Cycle
```

**EndFunction:**

```
RET 4
```

# Передача параметров с использованием STDCALL

**.data**

n **dd** 5

**.data?**

result **dd** ?

**.code**

**PUSH** n

**CALL** Factorial

**MOV** result, EAX

# Создание локальных переменных в стеке

- $[ESP]$  – точка возврата
  - $[ESP + 4]$  – первый параметр функции
  - $[ESP + 8]$  – второй параметр функции
- и т. д.

# Создание локальных переменных в стеке

## **SUB ESP, 4**

- [ESP] – локальная переменная
  - [ESP + 4] – точка возврата
  - [ESP + 8] – первый параметр функции
  - [ESP + 12] – второй параметр функции
- и т. д.

# Пролог функции

**PUSH EBP**

**MOV EBP, ESP**

**SUB ESP, 8**

- [EBP] – исходное значение EBP
  - [EBP + 4] – точка возврата
  - [EBP + 8] – первый параметр функции
  - [EBP + 12] – второй параметр функции
- и т. д.
- [EBP – 4] – вторая локальная переменная
  - [EBP – 8] – первая локальная переменная
- и т. д.





# Эпилог функции

**MOV ESP, EBP**

**POP EBP**

**RET 12**