**Федеральное государственное образовательное учреждение   
высшего образования**

**«ОРЛОВСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ   
ИМЕНИ И.С. ТУРГЕНЕВА»**

**Институт приборостроения, автоматизации и информационных технологий**

**Кафедра информационных систем и цифровых технологий**

Отчет к лабораторной работе № 8

«Использование стека и рекурсивных процедур.   
Организация передачи параметров через стек   
в процедуру вычисления факториала числа»

Выполнила:

Студентка группы 21ПГ

Банных Мария Алексеевна

Приняла:

доцент кафедры   
информационных систем   
и цифровых технологий  
Конюхова Оксана Владимировна  
  
доцент кафедры   
информационных систем   
и цифровых технологий  
Амелина Ольга Викторовна

Орёл 2022 год

Ответы на контрольные вопросы

1. *Какие регистры используются при работе со стеком? Назначение регистра BP.*

При работе со стеком используются регистры:

* ss – сегментный регистр стека
* sр/еsр – регистр указателя стека
* bр/еbр – регистр указателя базы кадра стека

Регистр *BP* – указатель базы кадра стека. Например, при входе в процедуру выполняется передача нужных параметров путем записи их в стек. Если процедура также использует стек, то доступ к этим параметрам становится проблематичным. Выход заключается в том, чтобы после записи параметров в регистр *BP* записать адрес вершины стека *SP*. Значение регистра *SP* в дальнейшем будет изменяться, однако в регистре *BP* хранится адрес, используя который, можно получить доступ к переданным параметрам.

1. *Когда передаваемые в процедуру аргументы записываются в стек? Какое место они занимают в стеке после входа в процедуру?*

Стек может использоваться для временного хранения значений регистров и ячеек памяти, адресов возврата из процедур, передачи параметров в процедуры.

При передаче управления процедуре процессор автоматически записывает в вершину стека два (для процедур ближнего вызова) или четыре (для процедур дальнего вызова) байта – *адрес возврата* в вызывающую программу. Если предварительно в стек были записаны переданные процедуре параметры или указатели на них, то они окажутся под адресом возврата. Информация записывается и считывается по принципу «последним записан – первым считан» (*LIFO – Last Input First Output*).

1. *Формат процедуры при использовании передачи параметров через стек. Пролог и эпилог процедуры.*

Команда PUSH имеет один операнд, который может быть двухбайтовым регистром, кроме регистра *CS*, или ячейкой памяти такого же размера. При записи в стек данного сначала производится уменьшение на 2 содержимого указателя стека *SP* (стек оперирует словами), а затем по адресу, указываемому парой *SS:SP* , производится запись операнда - источника.

Команда *POP* также имеет один операнд, который может быть двухбайтовым регистром, кроме регистра *CS*, или ячейкой памяти. При считывании слова из стека в качестве адреса берётся текущее содержимое указателя стека в сегменте стека (*SS:SP*), а после извлечения данного слова в операнд - приёмник содержимое *SP* увеличивается на 2.

Для произвольного доступа к данным в стеке используется регистр *BP*. Для корректной работы с использованием этого регистра содержимое стека должно быть правильно проинициализировано, что предполагает формирование в нём адреса, который бы непосредственно указывал на переданные данные. Для этого в начала процедуры необходимо включить дополнительный фрагмент кода – ***пролог процедуры***. Конец процедуры также должен быть оформлен особым образом для обеспечения корректного возврата из процедуры. Фрагмент кода, выполняющий эти действия, называется ***эпилогом процедуры***. При этом нужно откорректировать содержимое стека, убрав из него ставшие ненужными аргументы, переданные и использованные в процедуре.

1. *Какие действия выполняются в вызывающей программе после возврата из процедуры и для чего?*

После того, как процедура выполнилась, необходимо очистить стек, вытолкнув из него параметры. Тут тоже существует 2 способа: стек может быть очищен самой процедурой или кодом, который эту процедуру вызывал. Для первого способа используется команда RET с одним операндом, который должен быть равен количеству байтов, выталкиваемых из стека.

Для второго способа нужно использовать команду RET без операндов. Стек восстанавливается после выполнения процедуры путём прибавления значения к SP. С помощью такого способа программируются процедуры с переменным количеством параметров. Процедура не знает, сколько ей будет передано параметров, поэтому очистка стека должна выполняться вызывающим кодом.

1. *Как вычисляется адрес требуемого параметра в стеке в процедурах ближнего вызова? Как вычислить адрес аргумента в процедуре дальнего вызова?*

При вызове дальней процедуры в стеке сохраняется не только смещение, но и сегмент для возврата. Существуют две разновидности инструкции ret: ret и retf. Инструкция retf извлекает из стека не только смещение, но и сегмент (в отличие от команды ret, достающей смещение). При передаче управления процедуре процессор автоматически записывает в вершину стека два (для процедур ближнего вызова) или четыре (для процедур дальнего вызова) байта – адрес возврата в вызывающую программу. Если предварительно в стек были записаны переданные процедуре параметры или указатели на них, то они окажутся под адресом возврата.

1. *Передача параметров по ссылке и ее особенности. Какой оператор используется для записи в регистр адреса данного, а не самого данного?*

При передаче аргументов по ссылке в вызываемой процедуре обрабатывается не копия, а оригинал передаваемых данных. Поэтому при изменении данных в вызываемой процедуре они автоматически изменяются и в вызывающей программе, поскольку изменения касаются одной области памяти.

Этот механизм удобен для передачи больших массивов данных и для тех случаев, когда процедура должна модифицировать параметры, хотя он и медленнее из-за того, что процедура будет выполнять дополнительные действия для получения значений параметров.

mov ax, offset value

call procedure

1. *Передача параметров по значению и ее особенности*

При передаче параметров через стек по значению на их размер накладываются ограничения, связанные с размерностью стека. Кроме того, в этом случае в вызываемой процедуре обрабатываются копии параметров. Таким образом, в рассмотренном примере значение по адресу aa в сегменте данных не изменится, то есть останется равным 10, независимо от выполняемых над этим значением действий в процедуре.

Например, если параметры передаются в регистрах:

mov ax,word ptr value ; сделать копию значения

call procedure ; вызвать процедуру

Практическая часть:

.model small

steck segment stack "stack"

db 100 dup('$')

steck ends

data segment

nomber1 dw 3h

nomber2 dw 3h

result1 dw (?)

result2 dw (?)

data ends

kod segment

assume SS: steck, DS: data, CS: kod

begin:

mov ax, data

mov ds, ax

xor ax, ax

mov ax, word ptr nomber1 ;сделать копию значения

push ax ;запись в стек аргумента

CALL factorial

mov result1, ax ;помещаем результат в переменную result1

pop ax ;очищаем стек, забирая аргумент в регистр ax

mov bx, offset nomber2 ;помещаем относительный адрес переменной в bx

mov ax, word ptr[bx] ;помещаем элемент по этому адресу в bx

push ax ;запись в стек аргумента

CALL factorial

mov result2, ax ;помещаем результат в переменную result1

pop ax ;очищаем стек, забирая аргумент в регистр ax

mov ax, result1 ;выводим result1 и result2 в регистры ax и bx

mov bx, result2

mov ah, 4ch

int 21h

;\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

;Процедура нахождения факториала

factorial proc near

push bp

mov bp, sp

mov ax, [bp+4] ;доступ к аргументу по адресу nomber1 для процедуры

mov cx, [bp+4]

dec cx

mov bx, [bp+4]

povtor:

dec bx

mul bx

LOOP povtor

mov sp, bp ;восстановление значения регистра sp

pop bp ;восстановление значения старого bp

ret

factorial endp

;Конец процедуры нахождения факториала

;\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

kod ends

end begin

ends