МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ ФГБОУ ВО «ОРЛОВСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ ИМЕНИ И.С.ТУРГЕНЕВА»

Кафедра «Программная инженерия»

**Отчет**

По лабораторной работе №7  
на тему:  
“ Использование стека. Проверка баланса расстановки скобок в строке ”

Работу выполнил:  
Студент группы 01-ИТ

Колчев Д.Н.

Проверил:  
Доцент кафедры  
программной инженерии

Конюхова О.В.

Орёл, 2020

1. **Структура программы**

s\_s segment

dw 20 dup('$')

s\_s ends

d\_s segment

string db '({[1+2]}){1\*2]]','$';

result db 0

d\_s ends

c\_s segment

assume ss:s\_s,ds:d\_s,cs:c\_s

begin: ; начало программы

mov ax,d\_s

mov ds,ax

xor ax,ax

lea si, string

;Ищем в строке скобку

search:

lodsb

;Проверка, это конец строки?

cmp al, '$'

je endString

;Это открывающая или закрывающая скобка?

;Это открывающие скобки?

cmp al, '('

je inStack

cmp al, '{'

je inStack

cmp al, '['

je inStack

;Это закрывающие скобки?

cmp al, ')'

je outStack

cmp al, '}'

je outStack

cmp al, ']'

je outStack

jmp search

;Помещаем скобку в Stack, увеличиваем счётчик

inStack:

inc cx

push ax

jmp search

;Выниманием из Stack скобку, проверяем пару

outStack:

;Была передана лишняя закрыв. скобка?

cmp cx, 0

je error3

dec cx

pop bx

;Вскрытая скобка закрыта верно?

cmp bl, '('

jne close1

cmp al, ')'

jne error1

jmp search

close1:

cmp bl, '['

jne close2

cmp al, ']'

jne error1

jmp search

close2:

cmp bl, '{'

cmp al, '}'

jne error1

jmp search

;Остались ли незакрытые скобки?

endString:

cmp cx, 0

jne error2

jmp exit

;Скобки остались, это ошибка №2

error2:

mov result, 2

jmp exit

;Лишняя скобка передана, ошибка №3

error3:

mov result, 3

jmp exit

;Закрывающая скобка несоответствует открывающей, ош №1

error1:

mov result, 1

jmp exit

;Пред-завершение. Каков результат программы?

exit:

cmp result, 1

jne enough

;Ищем нужную скобку для исправления ошибки №1

cmp bl, '('

jne next1

mov bl, ')'

jmp enough

next1:

cmp bl, '{'

jne next2

mov bl, '}'

jmp enough

next2:

cmp bl, '['

mov bl, ']'

jmp enough

enough:

mov dl, result

xor dx, dx

mov dl, bl

mov ah,4ch

int 21h

c\_s ends

end begin

1. **Контрольные вопросы:**
2. **Память с последовательным доступом. Виды памяти с последовательным доступом.**

Память с последовательным доступом строится либо с использованием продвижения данных по цепочке элементов (по подобию с регистрами сдвига), либо с хранением данных в адресном ЗУ при соответствующем управлении адресом доступа.

Виды:

• видеопамять,

• буфер FIFO,

• стек.

1. **Инициализация полей структуры в программе.**

***Стек*** – это однонаправленная очередь, данные в которую помещаются и извлекаются в строго определённом порядке. Стековая память обеспечивает такой режим работы, когда информация записывается и считывается по принципу «последним записан – первым считан» (LIFO – Last Input First Output). Такая память используется для временного хранения данных, например, для запоминания и восстановления регистров процессора (контекста) при обработке подпрограмм и прерываний.

Под стек выделяется отдельный сегмент – сегмент стека, начальный адрес которого помещается в соответствующий сегментный регистр – SS. Адресация стека обеспечивается специальным регистром – указателем стека SP, в который предварительно помещается наибольший адрес области основной памяти, отведённой под стек (дно стека). Адрес последнего включённого в стек элемента называется ***вершиной стека*** (TOS – Top Of Stack).

1. **Режимы адресации для доступа к элементам структуры, для доступа к элементам массива структуру.** Для работы со стеком существуют две основные операции:

1) добавление элемента в вершину стека (PUSH);

2) извлечение элемента из вершины стека (POP).

Команда PUSH имеет один операнд, который может быть непосредственным значением, 2-байтовым регистром или адресом ячейки памяти. При записи в стек данного сначала производится уменьшение на 2 содержимого указателя стека SP (стек оперирует словами), которое затем используется в качестве адреса ячейки, куда и производится запись. Команда POP также имеет один операнд, который может быть 2-байтовым регистром. При считывании слова из стека в качестве адреса этого данного берётся текущее содержимое указателя стека, а после извлечения данного содержимое SP увеличивается на 2.

1. **Какие регистры используются при работе со стеком? Каково их назначение?**

Под стек выделяется отдельный сегмент – сегмент стека, начальный адрес которого помещается в соответствующий сегментный регистр – SS*(Stack Segment)*. Адресация стека обеспечивается специальным регистром – указателем стека SP, в который предварительно помещается наибольший адрес области основной памяти, отведённой под стек (дно стека).

Для доступа к элементам не в вершине, а внутри стека используется 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. **Как изменяется содержимое указателя вершины стека при включении нового элемента в стек и извлечении элемента из стека?**

При помещении элементов в стек адрес вершины стека (содержимое регистра SP) уменьшается (смещается в сторону меньших адресов), а при извлечении элементов из стека – увеличивается (смещается в сторону больших адресов).