**《软件逆向工程》实验报告**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **年级、专业、班级** | | **2022级信息安全02班** | | | **姓名** | **姚凡** |
| **实验题目** | **X86汇编子程序** | | | | | |
| **实验时间** | **2024/10/12** | | **实验地点** | **DS3401** | | |
| **实验成绩** |  | | **实验性质** | **□验证性 □设计性 ☑综合性** | | |
| 教师评价：  ☑算法/实验过程正确； ☑源程序/实验内容提交； ☑程序结构/实验步骤合理；  □实验结果正确； □语法、语义正确； ☑报告规范；  其他：  评价教师签名： 杨吉云 | | | | | | |
| 1. 实验目的 2. 验证X86汇编子程序结构 3. 验证X86汇编子程序参数传递方式运算 4. 验证X86汇编子程序运行时堆栈的变化 | | | | | | |
| 1. 实验项目内容 2. 编写两个数的加法子程序，参数传递方式为寄存器传参、变量传参数、变量地址传参数，在主程序传递参数调用子程序，记录堆栈变化情况。 3. 编写子程序实现对数组求和、平均值，在主程序中传递数组地址和数组个数调用子程序，记录堆栈变化情况。 4. 编写子程序将其中字母大小写反转、将其中的数字转换为二进制存储到数组中。主程序定义字符串为“asdterDXRT345\*!@as4”，以及存储数组，通过传递字符串地址和数组地址调用子程序，记录堆栈变化情况. | | | | | | |
| 1. **实验过程或算法（源程序）** 2. **加法子程序实现**   **（1）寄存器传参**  .386  .model flat,stdcall  option casemap:none  .stack 4096  includelib user32.lib  includelib kernel32.lib  includelib masm32.lib  ExitProcess PROTO,dwExitCode:DWORD  ;寄存器传参  .data  sum dd ?  a dd 1  b dd 2    .code  AddTwo proc    add eax,ebx  ret  AddTwo endp  main proc    mov eax,a  mov ebx,b  call AddTwo  mov sum,eax  invoke ExitProcess,0  ret  main endp  end main  **（2）变量传参**  .386  .model flat,stdcall  option casemap:none  .stack 4096  includelib user32.lib  includelib kernel32.lib  includelib masm32.lib  ExitProcess PROTO,dwExitCode:DWORD  ;变量传参  .data  sum dw ?  a dw 1  b dw 2    .code  AddTwo proc    push ebp  mov ebp,esp  mov ax,[ebp+12]  mov bx,[ebp+8]  add ax,bx  pop ebp  ret  AddTwo endp  main proc    push a  push b  call AddTwo  mov sum,ax  pop a  pop b  ret  main endp  end main  **（3）变量地址传参**  .386  .model flat,stdcall  option casemap:none  .stack 4096  includelib user32.lib  includelib kernel32.lib  includelib masm32.lib  ExitProcess PROTO,dwExitCode:DWORD  ;变量地址传参  .data  sum dw ?  a dw 1  b dw 2    .code  AddTwo proc    push ebp  mov ebp,esp  mov eax,[ebp+12]  mov ax,[eax]  mov ebx,[ebp+8]  mov bx,[ebx]  add ax,bx  pop ebp  ret  AddTwo endp  main proc    push offset a  push offset b  call AddTwo  mov sum,ax  ret  main endp  end main   1. **编写子程序实现对数组求和、平均值，在主程序中传递数组地址和数组个数调用子程序，记录堆栈变化情况**   .386  .model flat,stdcall  option casemap:none  .stack 4096  includelib user32.lib  includelib kernel32.lib  includelib masm32.lib  ExitProcess PROTO, dwExitCode:DWORD  .data  arrayA dd 1h, 2h, 3h, 4h, 5h ; 定义整数数组  sum dd ? ; 存储数组的总和  ave dd ? ; 存储数组的平均值  .code  addArray proc  push ebp ; 保存基指针  mov ebp, esp ; 建立新的基指针  mov esi, [ebp + 12] ; 获取数组地址  mov ecx, [ebp + 8] ; 获取数组长度  xor eax, eax ; 将 eax 清零，用于存储总和  L1:  add eax, [esi] ; 将当前数组元素加到总和中  add esi, 4 ; 指向下一个数组元素  loop L1 ; 循环，直到 ecx 为 0  mov sum, eax ; 将总和存储到 sum 变量中  pop ebp ; 恢复基指针  ret ; 返回  addArray endp  averageArray proc  push ebp ; 保存基指针  mov ebp, esp ; 建立新的基指针  mov eax, [ebp + 12] ; 获取总和  mov ebx, [ebp + 8] ; 获取数组长度  xor edx, edx ; 清除 edx，准备进行除法  div ebx ; 执行除法，结果存入 eax，余数存入 edx  mov ave, eax ; 将平均值存储到 ave 变量中  pop ebp ; 恢复基指针  ret ; 返回  averageArray endp  main proc  push offset arrayA  push lengthof arrayA ; 然后推送数组的长度  call addArray ; 调用数组求和子程序  push sum ; 先推送求和结果的地址  push lengthof arrayA ; 再推送数组的长度  call averageArray ; 调用平均值计算子程序  invoke ExitProcess, 0 ; 结束程序  ret  main endp  end main   1. **编写子程序将其中字母大小写反转、将其中的数字转换为二进制存储到数组中。主程序定义字符串为“asdterDXRT345\*!@as4”，以及存储数组，通过传递字符串地址和数组地址调用子程序，记录堆栈变化情况**   .386  .model flat,stdcall  option casemap:none  .stack 4096  includelib user32.lib  includelib kernel32.lib  includelib masm32.lib  ExitProcess PROTO,dwExitCode:DWORD  .data  strA byte "asdterDXRT345\*!@as4",0  arrayB dword 4 DUP(0)  .code  Switch proc  push esi  L1:  mov al,[esi]  cmp al,'0'  jb next2  cmp al,':'  jb next1  cmp al,'A'  jb next2  cmp al,'['  jb ChangeToLower  ;cmp al,'a'  ;jb next2  cmp al,'{'  jb ChangeToCapital  ;cmp al,'z'  ;ja next2  next1:  ;将数字存入新的数组  sub al,'0'  mov [edi],al  add edi,TYPE arrayB  jmp next2      next2:  ;处理非数字非字母字符，直接保留在原字符串中  inc esi  jmp L1    ChangeToLower:  ;处理字母大写向小写反转的  or al,00100000b  mov [esi],al  jmp next2    ChangeToCapital:  ;处理字母小写向大写反转的  and al,11011111b  mov [esi],al  jmp next2    pop esi  ret    Switch endp  main proc  mov esi, offset strA  mov edi, offset arrayB  call Switch  main endp  end main | | | | | | |
| 1. **实验结果及分析和（或）源程序调试过程** 2. **实现加法子程序** 3. **寄存器传参**     **向EAX中传入参数a的值1，向EBX中传入参数b的值2**    **跳转到加法子过程**    **EAX的值更新为3，加法操作成功**    **堆栈变化情况,寄存器传参时只会将返回地址压入栈中，如下图所示：**    **结束退出:**     1. **变量传参**   **首先把变量a和变量b的地址分别传入寄存器EAX和EBX中**    **跳转到AddTwo子过程进行加法操作**    **由于使用的是变量传参的方式，需要先把ebp寄存器push进栈，然后使用ebp对变量进行寻址，然后把通过这个地址找到的变量值存到对应的寄存器中。如下图，EBP+12的位置是a变量的偏移地址，这时把[EBP+12]的值存入EAX中，所以EAX此时的值为00190001h；同理，EBX中存入了变量b的值**      **执行加法操作后，EAX中的值变为3，加法操作成功**    **堆栈变化情况,**  **变量传参时，首先把变量的值a, b依次压入栈**      **然后将调用子程序的返回地址压入栈**    **将EBP的值压入栈中**    **子过程结束，弹出EBP，EBP的值被恢复为栈顶值**    **弹出EBP，栈中已经没有EBP**     1. **变量地址传参**   **首先把两个变量的地址压入栈中，如下图所示**    **跳转到加法子过程，压入EBP，并把ESP的值赋给EBP**      **通过EBP进行寻址，找到变量a和变量b的地址，将这个地址存入EBX和EAX中**    **然后把EAX中的值（b的值）和EBX中的值（a的值）存入AX和BX中**      **加法操作执行后结果正确，如图所示**    **子过程结束，弹出EBP，EBP的值被恢复为栈顶值**    **最后,将ax的值放到sum中,结束**     1. **实现数组求和和求平均值子程序**   **初始化数组A为{1h,2h,3h,4h,5h}**    **求和结果为15(0000000Fh)**    **平均值结果为3(00000003h)**    **堆栈变化情况,**  **把数组的首地址压入栈中**    **把数组长度压入栈中**    **将求和子过程的返回地址压栈**    **EBP压入栈**    **EBP出栈，求和子过程地址出栈**    **sum变量入栈，数组长度入栈**    **数组平均值入栈**    **将求均值子过程的返回地址压栈**    **最后EBP压栈、出栈,结束**     1. **实现字符串大小写反转子程序**   **字符串大小写反转和数字保留到新数组中的过程在实验一中已经详细解释,这里就不再赘述**  **反转后的结果如图所示:**    **堆栈变化情况,**  **子程序返回地址入栈**    **原字符串压入栈**    **之后,根据代码逐个将原字符串中的字符进行判断修改,大小写字母互相反转,数字保存到新数组中,非字母非数字的字符串保留在原数组中,同时,栈中对应的字符也相应地被修改,如下图栈中原字符a反转为A**    **后面的小写字母同理转为大写字母**  **转后如图:**    **后面的大写字母同理转为小写字母**  **转后如图:**    **反转后的结果如图所示,可见栈中大小写字母已经全部反转:**    **将返回地址和字符串压入栈** | | | | | | |