FactorialRM.c

```
#include <stdio.h>
int factorial(int n);
void DumpS(int n);









int main()  {
   int n, f;









   printf("Enter an integer:");


   scanf("%d",&n);


   DumpS(64);


   f = factorial(n);


   DumpS(64);




   printf("%d! is %d\n",n,f);
}
```

FactorialRM.s

```
        .file   "FactorialRM.c"
        .section        .rodata
.LC0:
        .string "Enter an integer:"
.LC1:
        .string "%d"
.LC2:
        .string "%d! is %d\n"
        .text
        .globl  main
        .type   main, @function
main:
.LFB0:
        .cfi_startproc
        pushl   %ebp
        .cfi_def_cfa_offset 8
        .cfi_offset 5, -8
        movl    %esp, %ebp
        .cfi_def_cfa_register 5
        andl    $-16, %esp
        subl    $32, %esp
        movl    $.LC0, (%esp)
        call    printf
        leal    24(%esp), %eax
        movl    %eax, 4(%esp)
        movl    $.LC1, (%esp)
        call    __isoc99_scanf
        movl    $64, (%esp)
        call    DumpS
        movl    24(%esp), %eax
        movl    %eax, (%esp)
        call    factorial
        movl    %eax, 28(%esp)
        movl    $64, (%esp)
        call    DumpS
        movl    24(%esp), %eax
        movl    28(%esp), %edx
        movl    %edx, 8(%esp)
        movl    %eax, 4(%esp)
        movl    $.LC2, (%esp)
        call    printf
        leave
        .cfi_restore 5
        .cfi_def_cfa 4, 4
        ret
        .cfi_endproc
.LFE0:
        .size   main, .-main
        .ident  "GCC: (GNU) 4.8.5 20150623 (Red Hat
4.8.5-39)"
        .section        .note.GNU-stack,"",@progbits
```

Adjust the Stack Pointer to reserve space for main's data.

Pass the string and call printf.

Pass the two arguments and call scanf.

Pass 64 and call DumpS.

Pass n and call factorial.

Pass 64 and call DumpS.

Pass the three arguments and call printf.

FactorialRF.c

```
int factorial(int n) {


   int result;
   DumpS(64);

   if(n <= 1) {

      result = 1;

   } else {
      result = n * factorial(n-1);



   }
   DumpS(64);

   return result;
}
```

Push 64 as the Actual Argument Value, then call the subprogram.

Note that n, the Formal Argument, is at the positive offset 8 from the Frame Pointer.

Note that result, the Local Variable, is at the negative offset 12 from the Frame Pointer.

Moves result, the local variable, into %eax, which is the protocol used to "return" values to the caller.

FactorialRF.s

```
        .file   "FactorialRF.c"
        .text
        .globl  factorial
        .type   factorial, @function
factorial:
.LFB0:
        .cfi_startproc
        pushl   %ebp
        .cfi_def_cfa_offset 8
        .cfi_offset 5, -8
        movl    %esp, %ebp
        .cfi_def_cfa_register 5
        subl    $40, %esp
        movl    $64, (%esp)
        call    DumpS
        cmpl    $1, 8(%ebp)
        jg      .L2
        movl    $1, -12(%ebp)
        jmp     .L3
.L2:
        movl    8(%ebp), %eax
        subl    $1, %eax
        movl    %eax, (%esp)
        call    factorial
        imull   8(%ebp), %eax
        movl    %eax, -12(%ebp)
.L3:
        movl    $64, (%esp)
        call    DumpS
        movl    -12(%ebp), %eax
        leave
        .cfi_restore 5
        .cfi_def_cfa 4, 4
        ret
        .cfi_endproc
.LFE0:
        .size   factorial, .-factorial
        .ident  "GCC: (GNU) 4.8.5 20150623 (Red Hat 4.8.5-39)"
        .section        .note.GNU-stack,"",@progbits
```

Push the Frame Pointer onto the stack.

Move the Stack Pointer to become the Frame Pointer.

Effectively pushes 10 4 byte words onto the stack as workspace.

Subtracts 1 from n then pushes it as the Actual Argument to factorial.

Multiplies the value returned (in %eax) by n, placing the product in %eax, which is then moved to result.

```
AS02> cc -m32 FactorialRM.c -S
AS02> cc -m32 FactorialRF.c -S
AS02> cc -m32 FactorialRM.s FactorialRF.s DumpS32.s -o FactorialRM
```

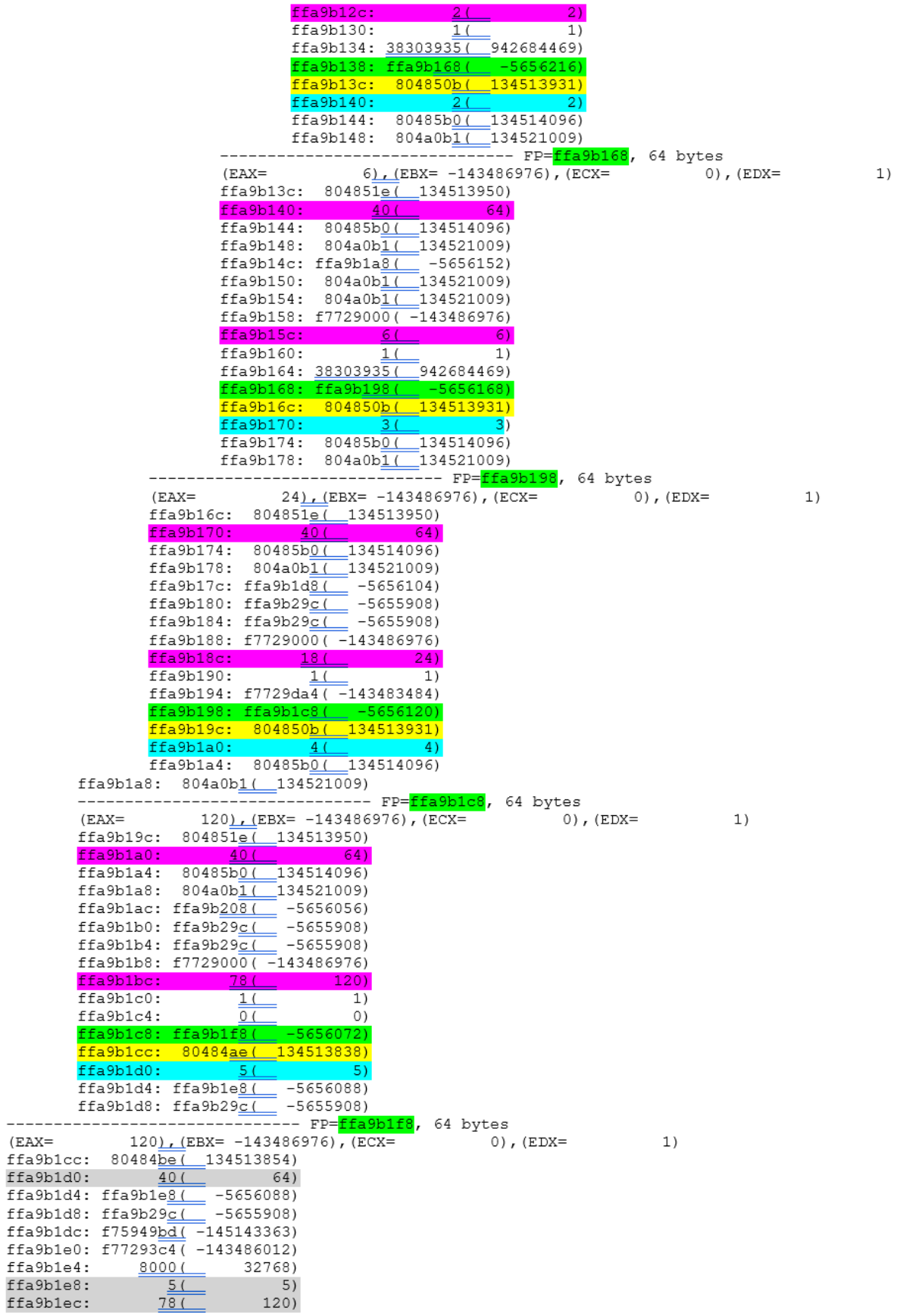
Compile `main` to obtain the corresponding assembly for the 32 bit architecture, and compile the function similarly. The compile the three assembly language components to create the executable.

```
AS02> ./FactorialRM
Enter an integer:5
------------------------------- FP=ffa9b1f8, 64 bytes
(EAX=          1),(EBX= -143486976),(ECX=          0),(EDX=          1)
ffa9b1cc:  80484a2(  134513826)
ffa9b1d0:       40(         64)
ffa9b1d4: ffa9b1e8(  -5656088)
ffa9b1d8: ffa9b29c(  -5655908)
ffa9b1dc: f75949bd( -145143363)
ffa9b1e0: f77293c4( -143486012)
ffa9b1e4:     8000(      32768)
ffa9b1e8:        5(          5)
ffa9b1ec: f7729000( -143486976)
ffa9b1f0:  80485e0(  134514144)
ffa9b1f4:        0(          0)
ffa9b1f8:        0(          0)
ffa9b1fc: f757c2a3( -145243485)
ffa9b200:        1(          1)
ffa9b204: ffa9b294(  -5655916)
ffa9b208: ffa9b29c(  -5655908)
        ------------------------------- FP=ffa9b1c8, 64 bytes
        (EAX=          5),(EBX= -143486976),(ECX=          0),(EDX=          1)
        ffa9b19c:  80484ee(  134513902)
        ffa9b1a0:       40(         64)
        ffa9b1a4:  80485b0(  134514096)
        ffa9b1a8:  804a0b1(  134521009)
        ffa9b1ac: ffa9b208(  -5656056)
        ffa9b1b0: ffa9b29c(  -5655908)
        ffa9b1b4: ffa9b29c(  -5655908)
        ffa9b1b8: f7729000( -143486976)
        ffa9b1bc:        0(          0)
        ffa9b1c0:        1(          1)
        ffa9b1c4:        0(          0)
        ffa9b1c8: ffa9b1f8(  -5656072)
        ffa9b1cc:  80484ae(  134513838)
        ffa9b1d0:        5(          5)
        ffa9b1d4: ffa9b1e8(  -5656088)
        ffa9b1d8: ffa9b29c(  -5655908)
                ------------------------------- FP=ffa9b198, 64 bytes
                (EAX=          4),(EBX= -143486976),(ECX=          0),(EDX=          1)
                ffa9b16c:  80484ee(  134513902)
                ffa9b170:       40(         64)
                ffa9b174:  80485b0(  134514096)
                ffa9b178:  804a0b1(  134521009)
                ffa9b17c: ffa9b1d8(  -5656104)
                ffa9b180: ffa9b29c(  -5655908)
                ffa9b184: ffa9b29c(  -5655908)
                ffa9b188: f7729000( -143486976)
                ffa9b18c:        0(          0)
                ffa9b190:        1(          1)
                ffa9b194: f7729da4( -143483484)
                ffa9b198: ffa9b1c8(  -5656120)
                ffa9b19c:  804850b(  134513931)
                ffa9b1a0:        4(          4)
                ffa9b1a4:  80485b0(  134514096)
                ffa9b1a8:  804a0b1(  134521009)
                        ------------------------------- FP=ffa9b168, 64 bytes
                        (EAX=          3),(EBX= -143486976),(ECX=          0),(EDX=          1)
                        ffa9b13c:  80484ee(  134513902)
                        ffa9b140:       40(         64)
                        ffa9b144:  80485b0(  134514096)
                        ffa9b148:  804a0b1(  134521009)
                        ffa9b14c: ffa9b1a8(  -5656152)
                        ffa9b150:  804a0b1(  134521009)
                        ffa9b154:  804a0b1(  134521009)
```

Color Legend:

- FP address
- Return address
- Arguments
- Recursive local data
- `main`'s data

```
ffa9b158: f7729000( -143486976)
ffa9b15c:         0(          0)
ffa9b160:         1(          1)
ffa9b164: 38303935(  942684469)
ffa9b168: ffa9b198(  -5656168)
ffa9b16c:  804850b(  134513931)
ffa9b170:         3(          3)
ffa9b174:  80485b0(  134514096)
ffa9b178:  804a0b1(  134521009)
        -------------------------------- FP=ffa9b138, 64 bytes
        (EAX=          2),(EBX= -143486976),(ECX=          0),(EDX=          1)
        ffa9b10c:  80484ee(  134513902)
        ffa9b110:        40(         64)
        ffa9b114:  80485b0(  134514096)
        ffa9b118:  804a0b1(  134521009)
        ffa9b11c: ffa9b178(  -5656200)
        ffa9b120:  804a0b1(  134521009)
        ffa9b124:  804a0b1(  134521009)
        ffa9b128: f7729000( -143486976)
        ffa9b12c:         0(          0)
        ffa9b130:         1(          1)
        ffa9b134: 38303935(  942684469)
        ffa9b138: ffa9b168(  -5656216)
        ffa9b13c:  804850b(  134513931)
        ffa9b140:         2(          2)
        ffa9b144:  80485b0(  134514096)
                ffa9b148:  804a0b1(  134521009)
                -------------------------------- FP=ffa9b108, 64 bytes
                (EAX=          1),(EBX= -143486976),(ECX=          0),(EDX=          1)
                ffa9b0dc:  80484ee(  134513902)
                ffa9b0e0:        40(         64)
                ffa9b0e4:  80485b0(  134514096)
                ffa9b0e8:  804a0b1(  134521009)
                ffa9b0ec: ffa9b148(  -5656248)
                ffa9b0f0:  804a0b1(  134521009)
                ffa9b0f4:  804a0b1(  134521009)
                ffa9b0f8: f7729000( -143486976)
                ffa9b0fc:         0(          0)
                ffa9b100:         1(          1)
                ffa9b104: 39303031(  959459377)
                ffa9b108: ffa9b138(  -5656264)
                ffa9b10c:  804850b(  134513931)
                ffa9b110:         1(          1)
                ffa9b114:  80485b0(  134514096)
                ffa9b118:  804a0b1(  134521009)
                -------------------------------- FP=ffa9b108, 64 bytes
                (EAX=          1),(EBX= -143486976),(ECX=          0),(EDX=          1)
                ffa9b0dc:  804851e(  134513950)
                ffa9b0e0:        40(         64)
                ffa9b0e4:  80485b0(  134514096)
                ffa9b0e8:  804a0b1(  134521009)
                ffa9b0ec: ffa9b148(  -5656248)
                ffa9b0f0:  804a0b1(  134521009)
                ffa9b0f4:  804a0b1(  134521009)
                ffa9b0f8: f7729000( -143486976)
                ffa9b0fc:         1(          1)
                ffa9b100:         1(          1)
                ffa9b104: 39303031(  959459377)
                ffa9b108: ffa9b138(  -5656264)
                ffa9b10c:  804850b(  134513931)
                ffa9b110:         1(          1)
                ffa9b114:  80485b0(  134514096)
                ffa9b118:  804a0b1(  134521009)
        -------------------------------- FP=ffa9b138, 64 bytes
        (EAX=          2),(EBX= -143486976),(ECX=          0),(EDX=          1)
        ffa9b10c:  804851e(  134513950)
        ffa9b110:        40(         64)
        ffa9b114:  80485b0(  134514096)
        ffa9b118:  804a0b1(  134521009)
        ffa9b11c: ffa9b178(  -5656200)
        ffa9b120:  804a0b1(  134521009)
        ffa9b124:  804a0b1(  134521009)
        ffa9b128: f7729000( -143486976)
```


ffa9b12c: 2(2)
ffa9b130: 1(1)
ffa9b134: 38303935(942684469)
ffa9b138: ffa9b168(-5656216)
ffa9b13c: 804850b(134513931)
ffa9b140: 2(2)
ffa9b144: 80485b0(134514096)
ffa9b148: 804a0b1(134521009)
---------- FP=ffa9b168, 64 bytes
(EAX= 6),(EBX= -143486976),(ECX= 0),(EDX= 1)
ffa9b13c: 804851e(134513950)
ffa9b140: 40(64)
ffa9b144: 80485b0(134514096)
ffa9b148: 804a0b1(134521009)
ffa9b14c: ffa9b1a8(-5656152)
ffa9b150: 804a0b1(134521009)
ffa9b154: 804a0b1(134521009)
ffa9b158: f7729000(-143486976)
ffa9b15c: 6(6)
ffa9b160: 1(1)
ffa9b164: 38303935(942684469)
ffa9b168: ffa9b198(-5656168)
ffa9b16c: 804850b(134513931)
ffa9b170: 3(3)
ffa9b174: 80485b0(134514096)
ffa9b178: 804a0b1(134521009)
---------- FP=ffa9b198, 64 bytes
(EAX= 24),(EBX= -143486976),(ECX= 0),(EDX= 1)
ffa9b16c: 804851e(134513950)
ffa9b170: 40(64)
ffa9b174: 80485b0(134514096)
ffa9b178: 804a0b1(134521009)
ffa9b17c: ffa9b1d8(-5656104)
ffa9b180: ffa9b29c(-5655908)
ffa9b184: ffa9b29c(-5655908)
ffa9b188: f7729000(-143486976)
ffa9b18c: 18(24)
ffa9b190: 1(1)
ffa9b194: f7729da4(-143483484)
ffa9b198: ffa9b1c8(-5656120)
ffa9b19c: 804850b(134513931)
ffa9b1a0: 4(4)
ffa9b1a4: 80485b0(134514096)
ffa9b1a8: 804a0b1(134521009)
---------- FP=ffa9b1c8, 64 bytes
(EAX= 120),(EBX= -143486976),(ECX= 0),(EDX= 1)
ffa9b19c: 804851e(134513950)
ffa9b1a0: 40(64)
ffa9b1a4: 80485b0(134514096)
ffa9b1a8: 804a0b1(134521009)
ffa9b1ac: ffa9b208(-5656056)
ffa9b1b0: ffa9b29c(-5655908)
ffa9b1b4: ffa9b29c(-5655908)
ffa9b1b8: f7729000(-143486976)
ffa9b1bc: 78(120)
ffa9b1c0: 1(1)
ffa9b1c4: 0(0)
ffa9b1c8: ffa9b1f8(-5656072)
ffa9b1cc: 80484ae(134513838)
ffa9b1d0: 5(5)
ffa9b1d4: ffa9b1e8(-5656088)
ffa9b1d8: ffa9b29c(-5655908)
---------- FP=ffa9b1f8, 64 bytes
(EAX= 120),(EBX= -143486976),(ECX= 0),(EDX= 1)
ffa9b1cc: 80484be(134513854)
ffa9b1d0: 40(64)
ffa9b1d4: ffa9b1e8(-5656088)
ffa9b1d8: ffa9b29c(-5655908)
ffa9b1dc: f75949bd(-145143363)
ffa9b1e0: f77293c4(-143486012)
ffa9b1e4: 8000(32768)
ffa9b1e8: 5(5)
ffa9b1ec: 78(120)

```
ffa9b1f0:  80485e0(  134514144)
ffa9b1f4:        0(          0)
ffa9b1f8:        0(          0)
ffa9b1fc: f757c2a3( -145243485)
ffa9b200:        1(          1)
ffa9b204: ffa9b294(  -5655916)
ffa9b208: ffa9b29c(  -5655908)
5! is 120
```