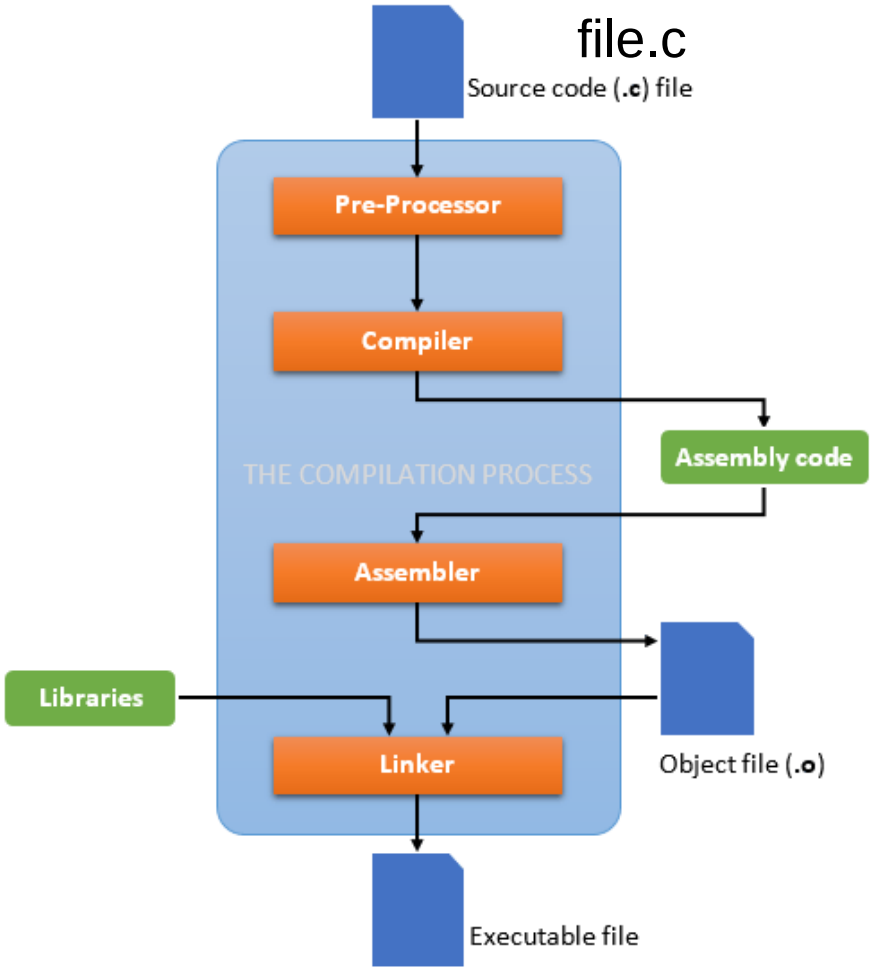
Disassembly

תרגול 7

ניתוח קוד

Reminder - Compilation Process



a.out / file (machine code)

Credit to: https://medium.com/@laura.derohan/compiling-c-files-with-gcc-step-by-step-8e78318052

How to - Disassembly of code

- Compilation of code:
  - gcc code.c
  - We get the file: a.out
- Disassembly:
  - objdump -d a.out
  - We get an assembly-like code that represents the c code appeared in file code.c
  - objdump –t a.out
  - This will print out the symbol table of the file. The symbol table includes the names of all functions and global variables in the file, the names of all the functions being called by the file, and their addresses.
  - strings a.out will print all strings in the code

Basic:

- Many times we work with an executive file and we are interested in the code that behind it.
- We can use the disassembly option or the debugger option in order to analyze the executive file and understand what is does.
- Sometimes we want to use both options.
- Disassembly enable us to get an assembly-like file that represent the activity of the executive file.

Important aspects

- In disassembly we only get the code of the functions in the files and functions that were used by the files.
- We don’t get the code of the system’s functions (printf, scanf…).
- We don’t get the values of global constants or strings.
- Many times there are optimizations or nops added by the compiler – what make is harder to understand. For example,
  - nop
  - xchg %cx, %cx

An example

- While using disassember there are many global general functions added (init, start) usually we don’t care about them.
- Show Tirgul6d_disass.txt

main:

```
0000000000400610 <main>:
  400610:       55                      push   %rbp
  400611:       48 89 e5                mov    %rsp,%rbp
  400614:       b8 00 00 00 00          mov    $0x0,%eax
  400619:       e8 9f ff ff ff          callq  4005bd <hello>
  40061e:       bf 04 00 00 00          mov    $0x4,%edi
  400623:       e8 a5 ff ff ff          callq  4005cd <even>
  400628:       b8 00 00 00 00          mov    $0x0,%eax
  40062d:       5d                      pop    %rbp
  40062e:       c3                      retq
  40062f:       90                      nop
```

main:

```
0000000000400610 <main>:
  400610:       55                      push   %rbp
  400611:       48 89 e5                mov    %rsp,%rbp
  400614:       b8 00 00 00 00          mov    $0x0,%eax
  400619:       e8 9f ff ff ff          callq  4005bd <hello>
  40061e:       bf 04 00 00 00          mov    $0x4,%edi
  400623:       e8 a5 ff ff ff          callq  4005cd <even>
  400628:       b8 00 00 00 00          mov    $0x0,%eax
  40062d:       5d                      pop    %rbp
  40062e:       c3                      retq
  40062f:       90                      nop
```

main:

```
0000000000400610 <main>:
  400610:       55                      push   %rbp
  400611:       48 89 e5                mov    %rsp,%rbp
  400614:       b8 00 00 00 00          mov    $0x0,%eax
  400619:       e8 9f ff ff ff          callq  4005bd <hello>
  40061e:       bf 04 00 00 00          mov    $0x4,%edi
  400623:       e8 a5 ff ff ff          callq  4005cd <even>
  400628:       b8 00 00 00 00          mov    $0x0,%eax
  40062d:       5d                      pop    %rbp
  40062e:       c3                      retq
  40062f:       90                      nop
```

main:

```
0000000000400610 <main>:
  400610:       55                      push   %rbp
  400611:       48 89 e5                mov    %rsp,%rbp
  400614:       b8 00 00 00 00          mov    $0x0,%eax
  400619:       e8 9f ff ff ff          callq  4005bd <hello>
  40061e:       bf 04 00 00 00          mov    $0x4,%edi
  400623:       e8 a5 ff ff ff          callq  4005cd <even>
  400628:       b8 00 00 00 00          mov    $0x0,%eax
  40062d:       5d                      pop    %rbp
  40062e:       c3                      retq
  40062f:       90                      nop
```

main:

```
0000000000400610 <main>:
  400610:       55                      push   %rbp
  400611:       48 89 e5                mov    %rsp,%rbp
  400614:       b8 00 00 00 00          mov    $0x0,%eax
  400619:       e8 9f ff ff ff          callq  4005bd <hello>
  40061e:       bf 04 00 00 00          mov    $0x4,%edi
  400623:       e8 a5 ff ff ff          callq  4005cd <even>
  400628:       b8 00 00 00 00          mov    $0x0,%eax
  40062d:       5d                      pop    %rbp
  40062e:       c3                      retq
  40062f:       90                      nop
```

hello:

```
0000000004005bd <hello>:
  4005bd:       55                      push   %rbp
  4005be:       48 89 e5                mov    %rsp,%rbp
  4005c1:       bf b4 06 40 00          mov    $0x4006b4,%edi
  4005c6:       e8 c5 fe ff ff          callq  400490 <puts@plt>
  4005cb:       5d                      pop    %rbp
  4005cc:       c3                      retq
```

- Address 0x4006b4 does not appear in the disassembly code we can see.
  - What does that tell us?
  - How can we find out what is its value?

hello:

```
0000000004005bd <hello>:
  4005bd:       55                      push   %rbp
  4005be:       48 89 e5                mov    %rsp,%rbp
  4005c1:       bf b4 06 40 00          mov    $0x4006b4,%edi
  4005c6:       e8 c5 fe ff ff          callq  400490 <puts@plt>
  4005cb:       5d                      pop    %rbp
  4005cc:       c3                      retq
```

- Function “puts” is a simplified version of the *printf*() *function.* It doesn’t have all printf formats and it always put the newline character in the end of its strings.

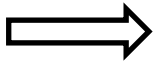
main:

```
0000000000400610 <main>:
  400610:       55                      push   %rbp
  400611:       48 89 e5                mov    %rsp,%rbp
  400614:       b8 00 00 00 00          mov    $0x0,%eax
  400619:       e8 9f ff ff ff          callq  4005bd <hello>
  40061e:       bf 04 00 00 00          mov    $0x4,%edi
  400623:       e8 a5 ff ff ff          callq  4005cd <even>
  400628:       b8 00 00 00 00          mov    $0x0,%eax
  40062d:       5d                      pop    %rbp
  40062e:       c3                      retq
  40062f:       90                      nop
```

even:

```
0000000004005cd <even>:
  4005cd:       55                      push   %rbp
  4005ce:       48 89 e5                mov    %rsp,%rbp
  4005d1:       48 83 ec 20             sub    $0x20,%rsp
  4005d5:       89 7d ec                mov    %edi,-0x14(%rbp)
  4005d8:       c7 45 fc 00 00 00 00    movl   $0x0,-0x4(%rbp)
  4005df:       eb 1d                   jmp    4005fe <even+0x31>
  4005e1:       8b 45 fc                mov    -0x4(%rbp),%eax
  4005e4:       8b 55 ec                mov    -0x14(%rbp),%edx
  4005e7:       01 d0                   add    %edx,%eax
  4005e9:       89 c6                   mov    %eax,%esi
  4005eb:       bf c1 06 40 00          mov    $0x4006c1,%edi
  4005f0:       b8 00 00 00 00          mov    $0x0,%eax
  4005f5:       e8 a6 fe ff ff          callq  4004a0 <printf@plt>
  4005fa:       83 45 fc 02             addl   $0x2,-0x4(%rbp)
  4005fe:       83 7d fc 09             cmpl   $0x9,-0x4(%rbp)
  400602:       7e dd                   jle    4005e1 <even+0x14>
  400604:       bf 0a 00 00 00          mov    $0xa,%edi
  400609:       e8 72 fe ff ff          callq  400480 <putchar@plt>
  40060e:       c9                      leaveq
  40060f:       c3                      retq
```

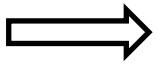
- What kind of a loop is it?



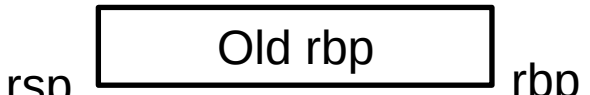
```
push   %rbp
mov    %rsp,%rbp
sub    $0x20,%rsp
mov    %edi,-0x14(%rbp)
movl   $0x0,-0x4(%rbp)
jmp    4005fe <even+0x31>
mov    -0x4(%rbp),%eax
mov    -0x14(%rbp),%edx
add    %edx,%eax
mov    %eax,%esi
mov    $0x4006c1,%edi
mov    $0x0,%eax
callq  4004a0 <printf@plt>
addl   $0x2,-0x4(%rbp)
cmpl   $0x9,-0x4(%rbp)
jle    4005e1 <even+0x14>
mov    $0xa,%edi
callq  400480 <putchar@plt
leaveq
retq
```

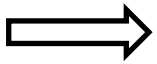
rsp

Old rbp

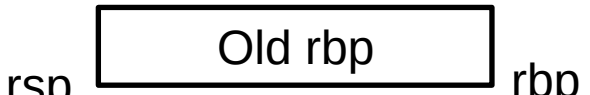


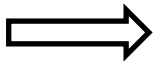
```
push   %rbp
mov    %rsp,%rbp
sub    $0x20,%rsp
mov    %edi,-0x14(%rbp)
movl   $0x0,-0x4(%rbp)
jmp    4005fe <even+0x31>
mov    -0x4(%rbp),%eax
mov    -0x14(%rbp),%edx
add    %edx,%eax
mov    %eax,%esi
mov    $0x4006c1,%edi
mov    $0x0,%eax
callq  4004a0 <printf@plt>
addl   $0x2,-0x4(%rbp)
cmpl   $0x9,-0x4(%rbp)
jle    4005e1 <even+0x14>
mov    $0xa,%edi
callq  400480 <putchar@plt
leaveq
retq
```



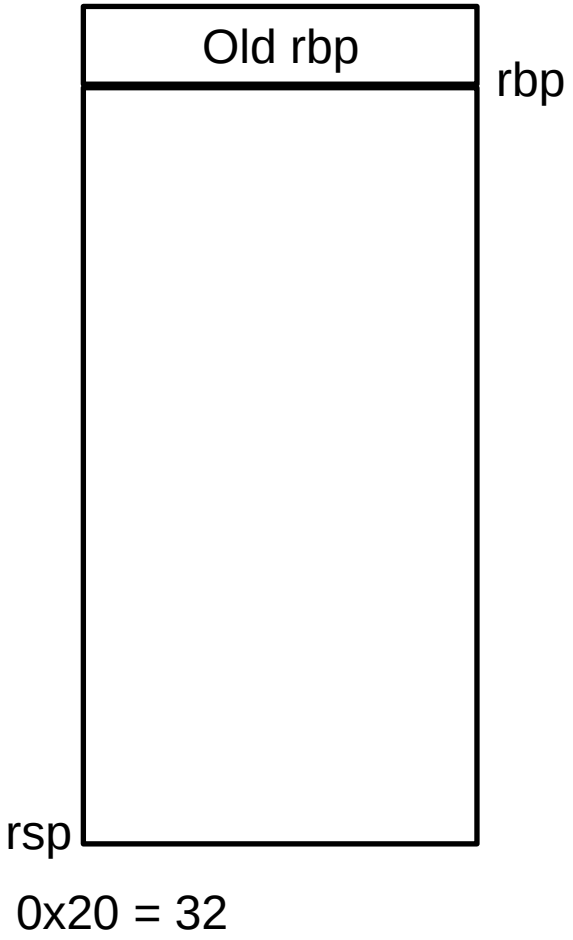


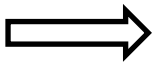
```
push   %rbp
mov    %rsp,%rbp
sub    $0x20,%rsp
mov    %edi,-0x14(%rbp)
movl   $0x0,-0x4(%rbp)
jmp    4005fe <even+0x31>
mov    -0x4(%rbp),%eax
mov    -0x14(%rbp),%edx
add    %edx,%eax
mov    %eax,%esi
mov    $0x4006c1,%edi
mov    $0x0,%eax
callq  4004a0 <printf@plt>
addl   $0x2,-0x4(%rbp)
cmpl   $0x9,-0x4(%rbp)
jle    4005e1 <even+0x14>
mov    $0xa,%edi
callq  400480 <putchar@plt
leaveq
retq
```



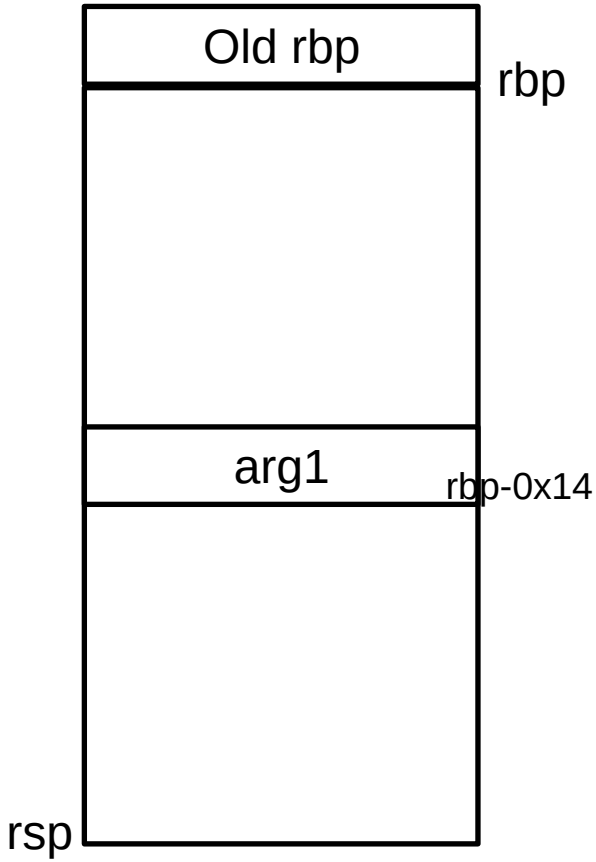


```
push   %rbp
mov    %rsp,%rbp
sub    $0x20,%rsp
mov    %edi,-0x14(%rbp)
movl   $0x0,-0x4(%rbp)
jmp    4005fe <even+0x31>
mov    -0x4(%rbp),%eax
mov    -0x14(%rbp),%edx
add    %edx,%eax
mov    %eax,%esi
mov    $0x4006c1,%edi
mov    $0x0,%eax
callq  4004a0 <printf@plt>
addl   $0x2,-0x4(%rbp)
cmpl   $0x9,-0x4(%rbp)
jle    4005e1 <even+0x14>
mov    $0xa,%edi
callq  400480 <putchar@plt
leaveq
retq
```

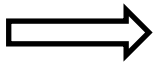




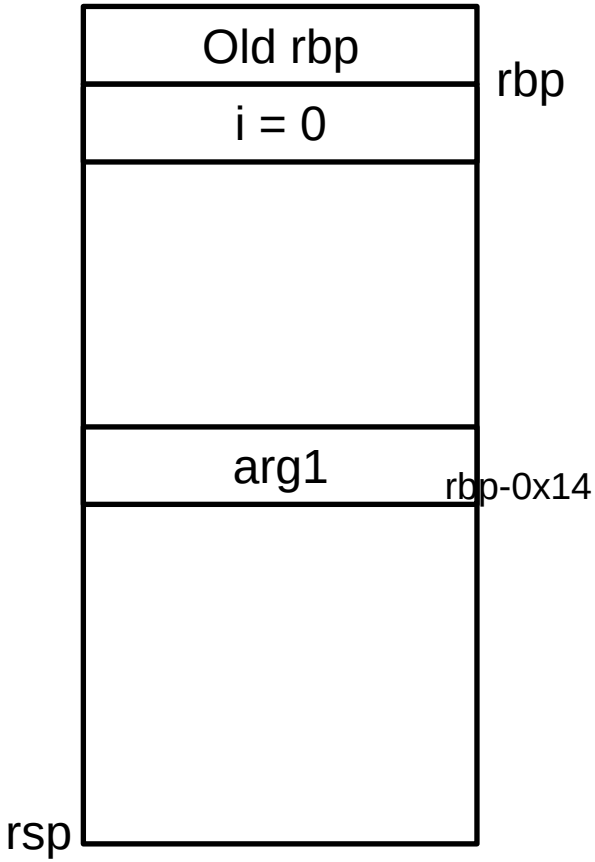
```
push   %rbp
mov    %rsp,%rbp
sub    $0x20,%rsp
mov    %edi,-0x14(%rbp)
movl   $0x0,-0x4(%rbp)
jmp    4005fe <even+0x31>
mov    -0x4(%rbp),%eax
mov    -0x14(%rbp),%edx
add    %edx,%eax
mov    %eax,%esi
mov    $0x4006c1,%edi
mov    $0x0,%eax
callq  4004a0 <printf@plt>
addl   $0x2,-0x4(%rbp)
cmpl   $0x9,-0x4(%rbp)
jle    4005e1 <even+0x14>
mov    $0xa,%edi
callq  400480 <putchar@plt>
leaveq
retq
```



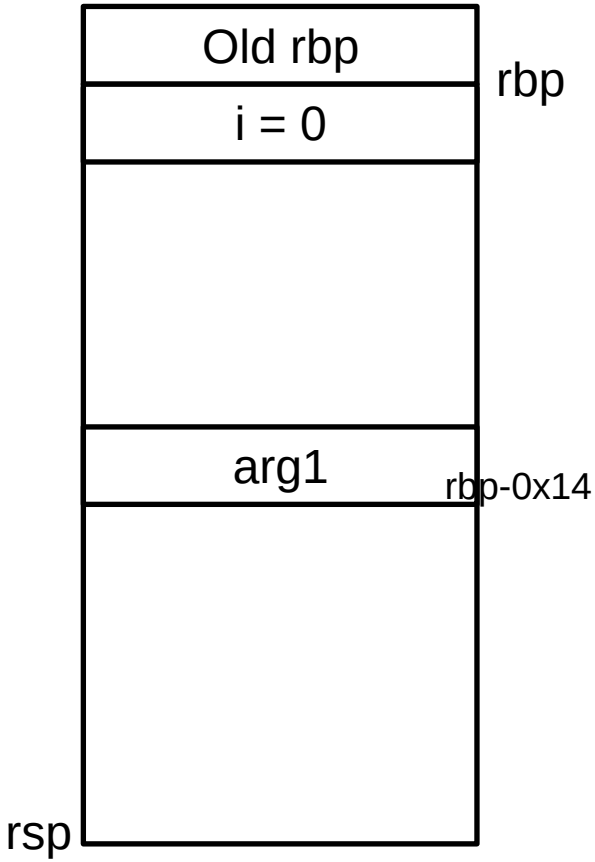
0x14 = 20



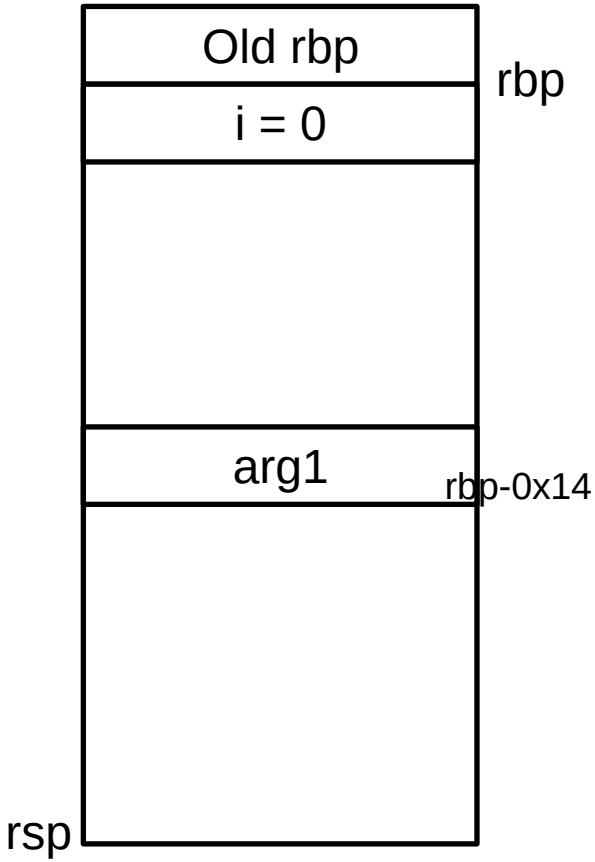
```
push   %rbp
mov    %rsp,%rbp
sub    $0x20,%rsp
mov    %edi,-0x14(%rbp)
movl   $0x0,-0x4(%rbp)
jmp    4005fe <even+0x31>
mov    -0x4(%rbp),%eax
mov    -0x14(%rbp),%edx
add    %edx,%eax
mov    %eax,%esi
mov    $0x4006c1,%edi
mov    $0x0,%eax
callq  4004a0 <printf@plt>
addl   $0x2,-0x4(%rbp)
cmpl   $0x9,-0x4(%rbp)
jle    4005e1 <even+0x14>
mov    $0xa,%edi
callq  400480 <putchar@plt
leaveq
retq
```



```
push   %rbp
mov    %rsp,%rbp
sub    $0x20,%rsp
mov    %edi,-0x14(%rbp)
movl   $0x0,-0x4(%rbp)
jmp    4005fe <even+0x31>
mov    -0x4(%rbp),%eax
mov    -0x14(%rbp),%edx
add    %edx,%eax
mov    %eax,%esi
mov    $0x4006c1,%edi
mov    $0x0,%eax
callq  4004a0 <printf@plt>
addl   $0x2,-0x4(%rbp)
cmpl   $0x9,-0x4(%rbp)
jle    4005e1 <even+0x14>
mov    $0xa,%edi
callq  400480 <putchar@plt
leaveq
retq
```

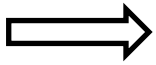


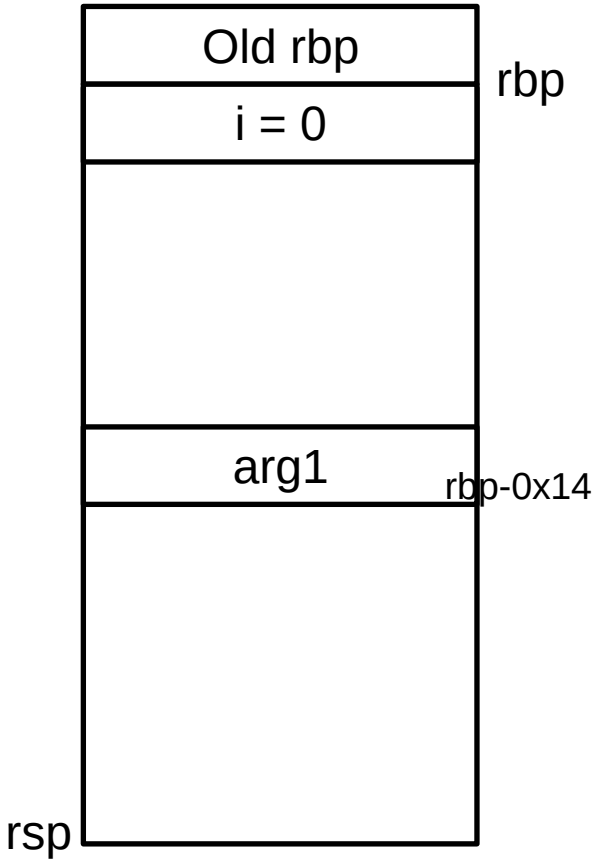
```
push   %rbp
mov    %rsp,%rbp
sub    $0x20,%rsp
mov    %edi,-0x14(%rbp)
movl   $0x0,-0x4(%rbp)
jmp    4005fe <even+0x31>
mov    -0x4(%rbp),%eax
mov    -0x14(%rbp),%edx
add    %edx,%eax
mov    %eax,%esi
mov    $0x4006c1,%edi
mov    $0x0,%eax
callq  4004a0 <printf@plt>
addl   $0x2,-0x4(%rbp)
cmpl   $0x9,-0x4(%rbp)
jle    4005e1 <even+0x14>
mov    $0xa,%edi
callq  400480 <putchar@plt
leaveq
retq
```



```
push   %rbp
mov    %rsp,%rbp
sub    $0x20,%rsp
mov    %edi,-0x14(%rbp)
movl   $0x0,-0x4(%rbp)
jmp    4005fe <even+0x31>
mov    -0x4(%rbp),%eax
mov    -0x14(%rbp),%edx
add    %edx,%eax
mov    %eax,%esi
mov    $0x4006c1,%edi
mov    $0x0,%eax
callq  4004a0 <printf@plt>
addl   $0x2,-0x4(%rbp)
cmpl   $0x9,-0x4(%rbp)
jle    4005e1 <even+0x14>
mov    $0xa,%edi
callq  400480 <putchar@plt
leaveq
retq
```

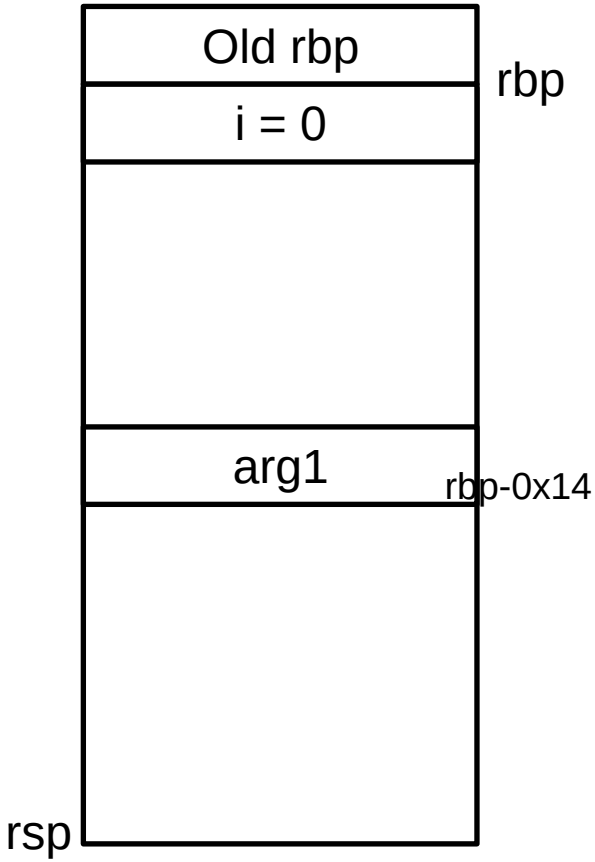
$i \leq 9$?





$eax \leftarrow i$

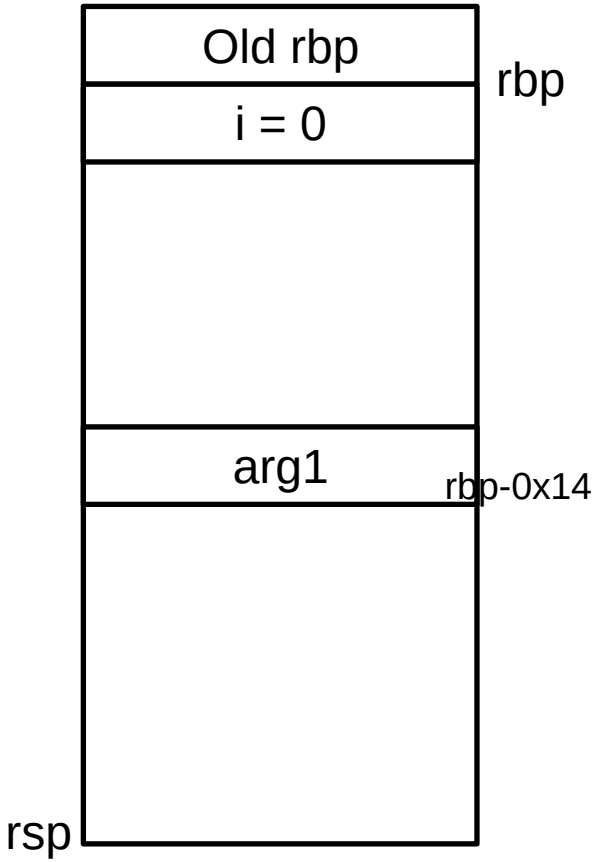
```
push   %rbp
mov    %rsp,%rbp
sub    $0x20,%rsp
mov    %edi,-0x14(%rbp)
movl   $0x0,-0x4(%rbp)
jmp    4005fe <even+0x31>
mov    -0x4(%rbp),%eax
mov    -0x14(%rbp),%edx
add    %edx,%eax
mov    %eax,%esi
mov    $0x4006c1,%edi
mov    $0x0,%eax
callq  4004a0 <printf@plt>
addl   $0x2,-0x4(%rbp)
cmpl   $0x9,-0x4(%rbp)
jle    4005e1 <even+0x14>
mov    $0xa,%edi
callq  400480 <putchar@plt
leaveq
retq
```



$eax \leftarrow i$

$edx \leftarrow arg1$

```
push   %rbp
mov    %rsp,%rbp
sub    $0x20,%rsp
mov    %edi,-0x14(%rbp)
movl   $0x0,-0x4(%rbp)
jmp    4005fe <even+0x31>
mov    -0x4(%rbp),%eax
mov    -0x14(%rbp),%edx
add    %edx,%eax
mov    %eax,%esi
mov    $0x4006c1,%edi
mov    $0x0,%eax
callq  4004a0 <printf@plt>
addl   $0x2,-0x4(%rbp)
cmpl   $0x9,-0x4(%rbp)
jle    4005e1 <even+0x14>
mov    $0xa,%edi
callq  400480 <putchar@plt
leaveq
retq
```

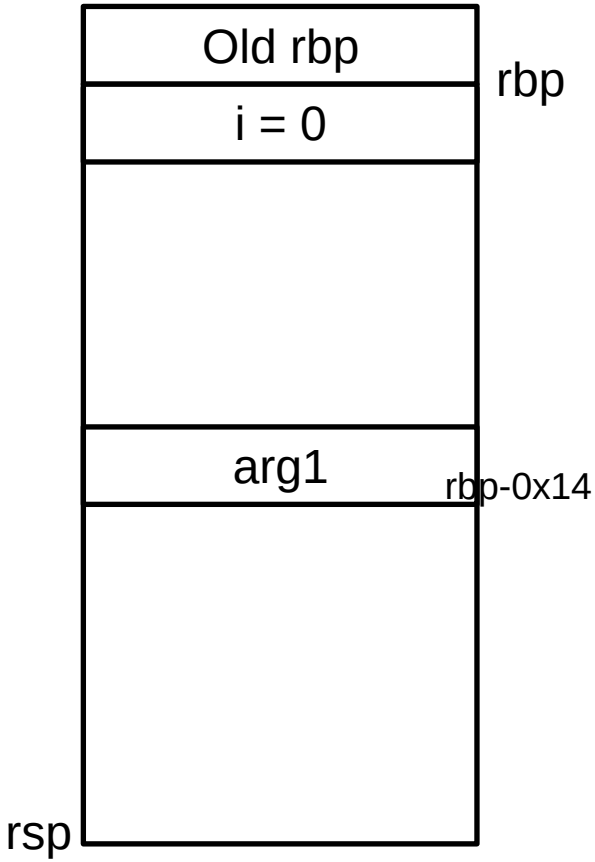


$eax \leftarrow i$

$edx \leftarrow arg1$

$eax \leftarrow arg1 + i$

```
push   %rbp
mov    %rsp,%rbp
sub    $0x20,%rsp
mov    %edi,-0x14(%rbp)
movl   $0x0,-0x4(%rbp)
jmp    4005fe <even+0x31>
mov    -0x4(%rbp),%eax
mov    -0x14(%rbp),%edx
add    %edx,%eax
mov    %eax,%esi
mov    $0x4006c1,%edi
mov    $0x0,%eax
callq  4004a0 <printf@plt>
addl   $0x2,-0x4(%rbp)
cmpl   $0x9,-0x4(%rbp)
jle    4005e1 <even+0x14>
mov    $0xa,%edi
callq  400480 <putchar@plt
leaveq
retq
```



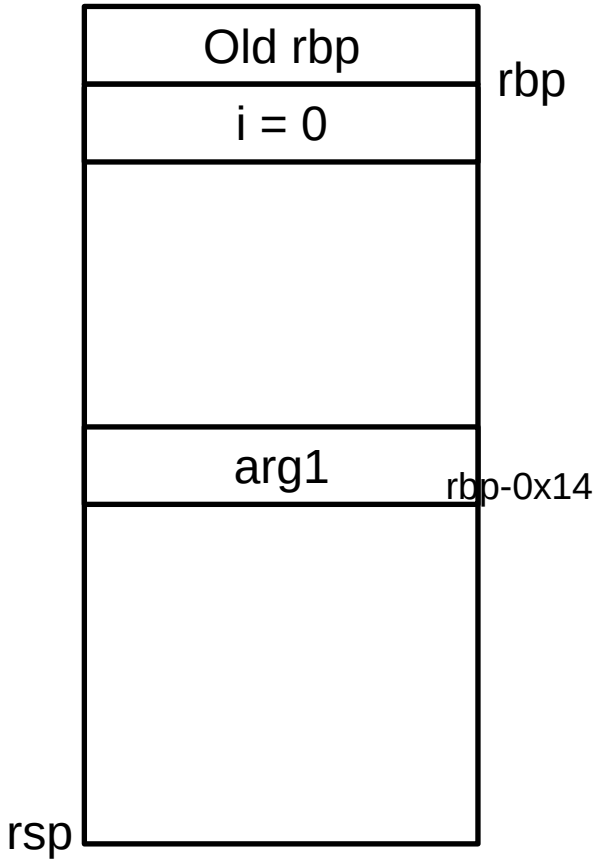
$eax \leftarrow i$

$edx \leftarrow arg1$

$eax \leftarrow arg1 + i$

$esi \leftarrow arg1 + i$

```
push   %rbp
mov    %rsp,%rbp
sub    $0x20,%rsp
mov    %edi,-0x14(%rbp)
movl   $0x0,-0x4(%rbp)
jmp    4005fe <even+0x31>
mov    -0x4(%rbp),%eax
mov    -0x14(%rbp),%edx
add    %edx,%eax
mov    %eax,%esi
mov    $0x4006c1,%edi
mov    $0x0,%eax
callq  4004a0 <printf@plt>
addl   $0x2,-0x4(%rbp)
cmpl   $0x9,-0x4(%rbp)
jle    4005e1 <even+0x14>
mov    $0xa,%edi
callq  400480 <putchar@plt
leaveq
retq
```



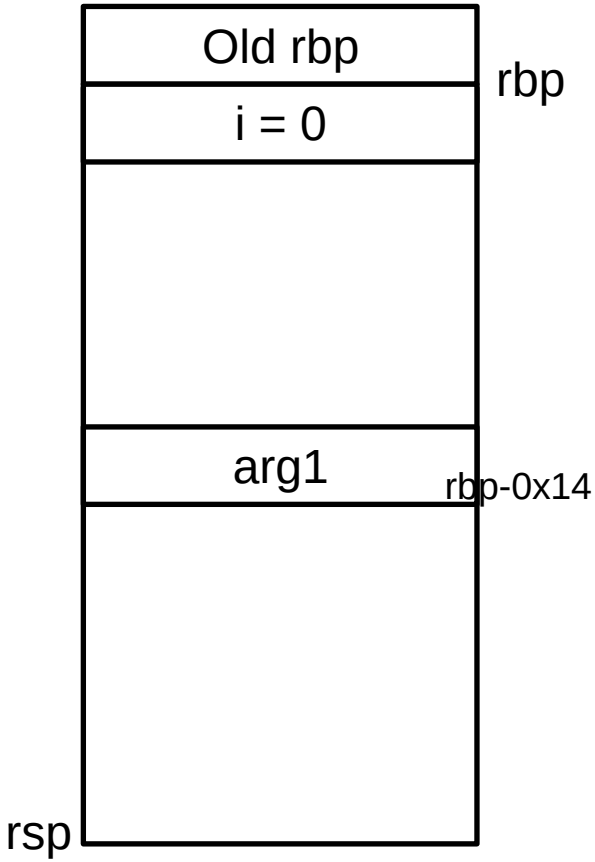
$eax \leftarrow i$

$edx \leftarrow arg1$

$eax \leftarrow arg1 + i$

$esi \leftarrow arg1 + i$

```
push   %rbp
mov    %rsp,%rbp
sub    $0x20,%rsp
mov    %edi,-0x14(%rbp)
movl   $0x0,-0x4(%rbp)
jmp    4005fe <even+0x31>
mov    -0x4(%rbp),%eax
mov    -0x14(%rbp),%edx
add    %edx,%eax
mov    %eax,%esi
mov    $0x4006c1,%edi
mov    $0x0,%eax
callq  4004a0 <printf@plt>
addl   $0x2,-0x4(%rbp)
cmpl   $0x9,-0x4(%rbp)
jle    4005e1 <even+0x14>
mov    $0xa,%edi
callq  400480 <putchar@plt
leaveq
retq
```



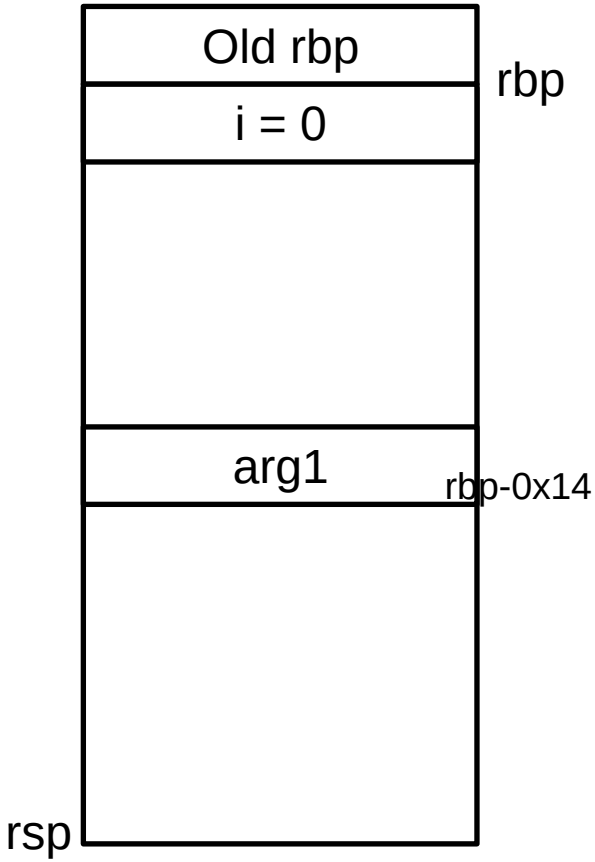
$eax \leftarrow i$

$edx \leftarrow arg1$

$eax \leftarrow arg1 + i$

$esi \leftarrow arg1 + i$

```
push   %rbp
mov    %rsp,%rbp
sub    $0x20,%rsp
mov    %edi,-0x14(%rbp)
movl   $0x0,-0x4(%rbp)
jmp    4005fe <even+0x31>
mov    -0x4(%rbp),%eax
mov    -0x14(%rbp),%edx
add    %edx,%eax
mov    %eax,%esi
mov    $0x4006c1,%edi
mov    $0x0,%eax
callq  4004a0 <printf@plt>
addl   $0x2,-0x4(%rbp)
cmpl   $0x9,-0x4(%rbp)
jle    4005e1 <even+0x14>
mov    $0xa,%edi
callq  400480 <putchar@plt>
leaveq
retq
```



$eax \leftarrow i$

$edx \leftarrow arg1$

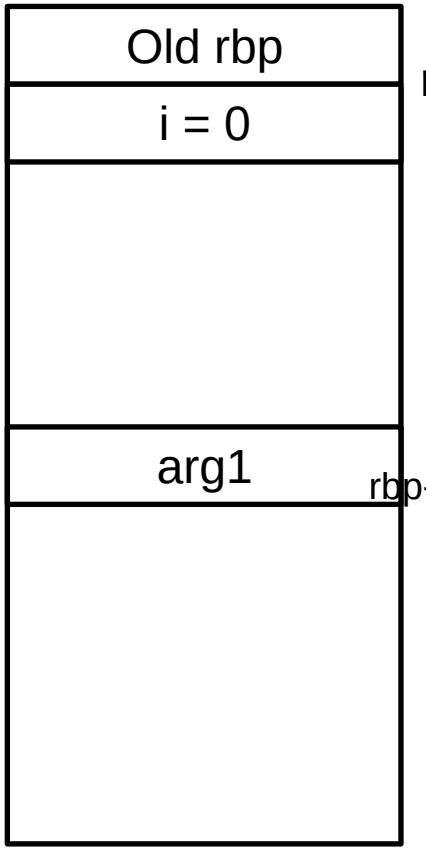
$eax \leftarrow arg1 + i$

$esi \leftarrow arg1 + i$

```
push   %rbp
mov    %rsp,%rbp
sub    $0x20,%rsp
mov    %edi,-0x14(%rbp)
movl   $0x0,-0x4(%rbp)
jmp    4005fe <even+0x31>
mov    -0x4(%rbp),%eax
mov    -0x14(%rbp),%edx
add    %edx,%eax
mov    %eax,%esi
mov    $0x4006c1,%edi
mov    $0x0,%eax
```

$printf("The\ number\ is\ \%d", arg1 + i);$

```
cmpl   $0x9,-0x4(%rbp)
jle    4005e1 <even+0x14>
mov    $0xa,%edi
callq  400480 <putchar@plt>
leaveq
retq
```



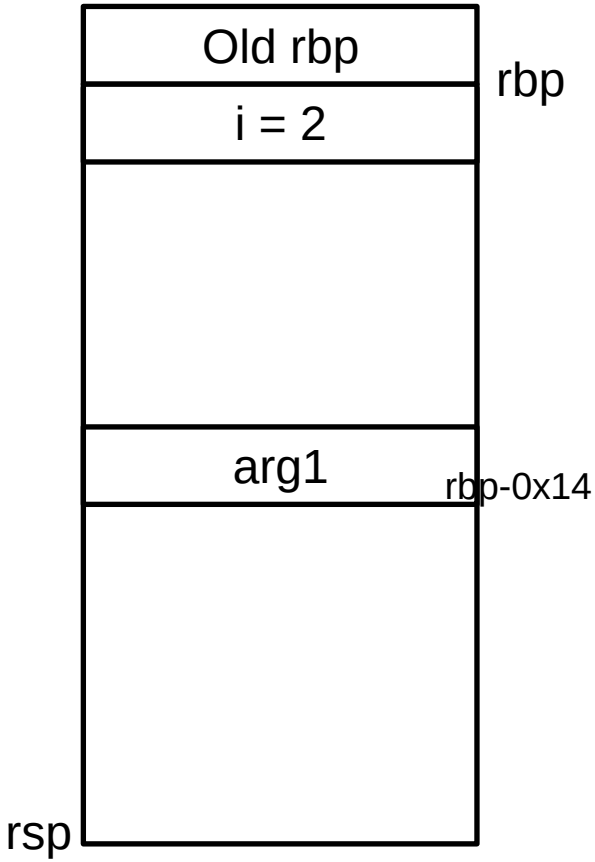
$eax \leftarrow i$

$edx \leftarrow arg1$

$eax \leftarrow arg1 + i$

$esi \leftarrow arg1 + i$

```
push   %rbp
mov    %rsp,%rbp
sub    $0x20,%rsp
mov    %edi,-0x14(%rbp)
movl   $0x0,-0x4(%rbp)
jmp    4005fe <even+0x31>
mov    -0x4(%rbp),%eax
mov    -0x14(%rbp),%edx
add    %edx,%eax
mov    %eax,%esi
mov    $0x4006c1,%edi
mov    $0x0,%eax
callq  4004a0 <printf@plt>
addl   $0x2,-0x4(%rbp)
cmpl   $0x9,-0x4(%rbp)
jle    4005e1 <even+0x14>
mov    $0xa,%edi
callq  400480 <putchar@plt
leaveq
retq
```



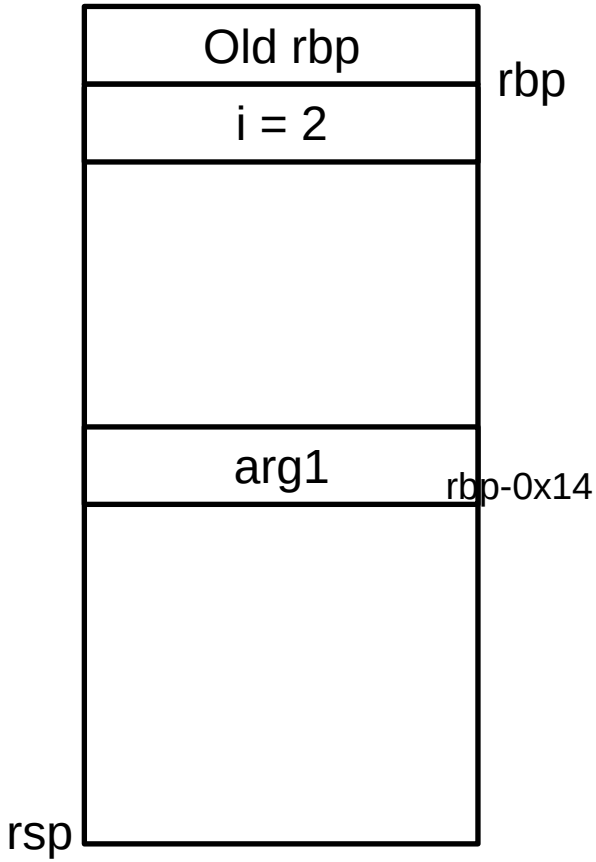
$eax \leftarrow i$

$edx \leftarrow arg1$

$eax \leftarrow arg1 + i$

$esi \leftarrow arg1 + i$

```
push   %rbp
mov    %rsp,%rbp
sub    $0x20,%rsp
mov    %edi,-0x14(%rbp)
movl   $0x0,-0x4(%rbp)
jmp    4005fe <even+0x31>
mov    -0x4(%rbp),%eax
mov    -0x14(%rbp),%edx
add    %edx,%eax
mov    %eax,%esi
mov    $0x4006c1,%edi
mov    $0x0,%eax
callq  4004a0 <printf@plt>
addl   $0x2,-0x4(%rbp)
cmpl   $0x9,-0x4(%rbp)
jle    4005e1 <even+0x14>
mov    $0xa,%edi
callq  400480 <putchar@plt
leaveq
retq
```



$eax \leftarrow i$

$edx \leftarrow arg1$

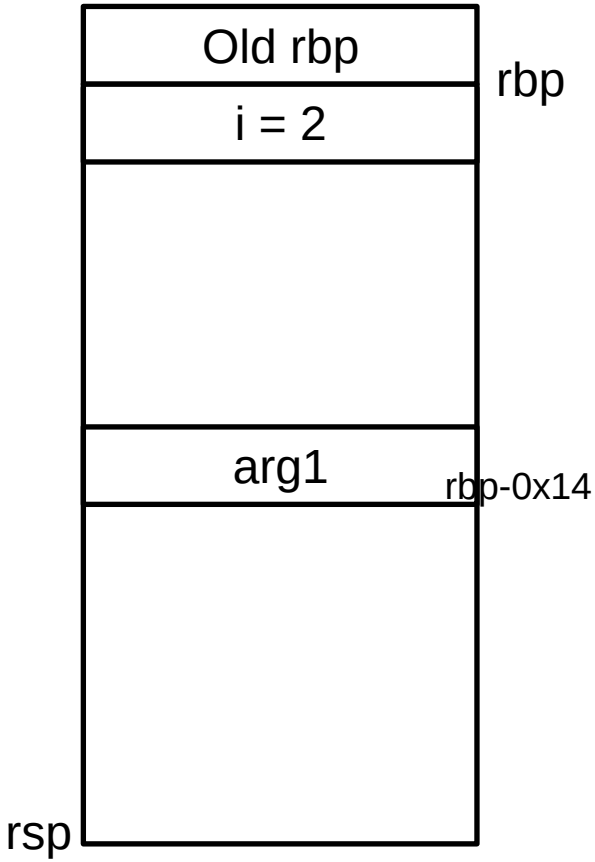
$eax \leftarrow arg1 + i$

$esi \leftarrow arg1 + i$

$i \leq 9?$

⇒

```
push   %rbp
mov    %rsp,%rbp
sub    $0x20,%rsp
mov    %edi,-0x14(%rbp)
movl   $0x0,-0x4(%rbp)
jmp    4005fe <even+0x31>
mov    -0x4(%rbp),%eax
mov    -0x14(%rbp),%edx
add    %edx,%eax
mov    %eax,%esi
mov    $0x4006c1,%edi
mov    $0x0,%eax
callq  4004a0 <printf@plt>
addl   $0x2,-0x4(%rbp)
cmpl   $0x9,-0x4(%rbp)
jle    4005e1 <even+0x14>
mov    $0xa,%edi
callq  400480 <putchar@plt
leaveq
retq
```



main:

```
0000000000400610 <main>:
  400610:       55                      push   %rbp
  400611:       48 89 e5                mov    %rsp,%rbp
  400614:       b8 00 00 00 00          mov    $0x0,%eax
  400619:       e8 9f ff ff ff          callq  4005bd <hello>
  40[illegible]e:       bf 04 00 00 00          mov    $0x4,%edi
  40            e8 a5 ff ff ff          callq  4005cd <even>
  4[illegible]  b8 00 00 00 00          mov    $0x0,%eax
  4006[illegible]                       pop    %rbp
  40062e:                               retq
  40062f:                               nop
```

Cleaning up the stack!

The C code:

```c
#include <stdio.h>

void hello() {
    printf("Hello there!\n");
}

void even(int a) {

    int i;
    for (i = 0 ; i < 10 ; i += 2) {
        printf("%d ", (a+i));
    }
    printf("\n");
}

int main() {

    hello();
    even(4);
    return 0;
}
```