```
08048500 <switchcode>:
 8048500:       push   %ebp
 8048501:       mov    %esp,%ebp
 8048503:       mov    0xc(%ebp),%ecx
 8048506:       xor    %eax,%eax
 8048508:       mov    0x8(%ebp),%edx
 804850b:       add    $0xfffffffec,%edx
 804850e:       cmp    $0x6,%edx
 8048511:       ja     8048540 <switchcode+0x40>
 8048513:       jmp    *0x80486e4(,%edx,4)
 804851a:       lea    0x0(%esi),%esi
 8048520:       mov    %ecx,%eax
 8048522:       imul   %ecx,%eax
 8048525:       jmp    8048545 <switchcode+0x45>
 8048527:       lea    (%ecx,%ecx,1),%eax
 804852a:       jmp    8048545 <switchcode+0x45>
 804852c:       lea    0x0(%esi,1),%esi
 8048530:       lea    0x4(%ecx),%eax
 8048533:       jmp    8048545 <switchcode+0x45>
 8048535:       mov    %ecx,%eax
 8048537:       neg    %eax
 8048539:       inc    %eax
 804853a:       jmp    8048545 <switchcode+0x45>
 804853c:       lea    0x0(%esi,1),%esi
 8048540:       mov    $0xf,%eax
 8048545:       mov    %ebp,%esp
 8048547:       pop    %ebp
```

```
 8048548:       ret
 8048549:       lea    0x0(%esi),%esi
```

```
(gdb) x/7w 0x80486e4

0x80486e4 <_fini+52>:   0x08048520
      0x08048527      0x08048530      0x08048530
0x80486f4 <_fini+68>:   0x08048540
      0x08048535      0x08048539
```

```
int switchcode(int x, int y)
{
	int result = 0;

	switch(x) {
	case 20:
		result = y*y;
		break;
	case 21:
		result = y*2;
		break;
	case 22:
	case 23:
		result = y+4;
		break;
	case 25:
		result = -y;
		/* Fall Through */
	case 26:
		result ++;
		break;
	default:
		result = 15;
	}
	return result;
}
```

- add $0xffffffec, %edx   -> edx -= 20

```
0804858c <recursive>:
 804858c:       push   %ebp
 804858d:       mov    %esp,%ebp
 804858f:       sub    $0x14,%esp
 8048592:       push   %ebx
 8048593:       mov    0x8(%ebp),%edx
 8048596:       mov    0xc(%ebp),%ebx
 8048599:       xor    %eax,%eax
 804859b:       test   $0x1,%dl
 804859e:       je     80485af <recursive+0x23>
 80485a0:       add    $0xfffffff8,%esp
 80485a3:       lea    0xffffffff(%edx),%eax
 80485a6:       push   %eax
 80485a7:       push   %ebx
 80485a8:       call   804858c <recursive>
 80485ad:       add    %ebx,%eax
 80485af:       mov    0xffffffe8(%ebp),%ebx
 80485b2:       mov    %ebp,%esp
 80485b4:       pop    %ebp
 80485b5:       ret
```

```
 int recursive(int x, int y)
{
    int result = 0;

    if (x & 0x1)
          result = y + recursive(y,x-1);

    return result;
 }
```

- mov instruction for recovery of %ebx (instead of pop)
- allocation of 20 bytes of unused space on stack