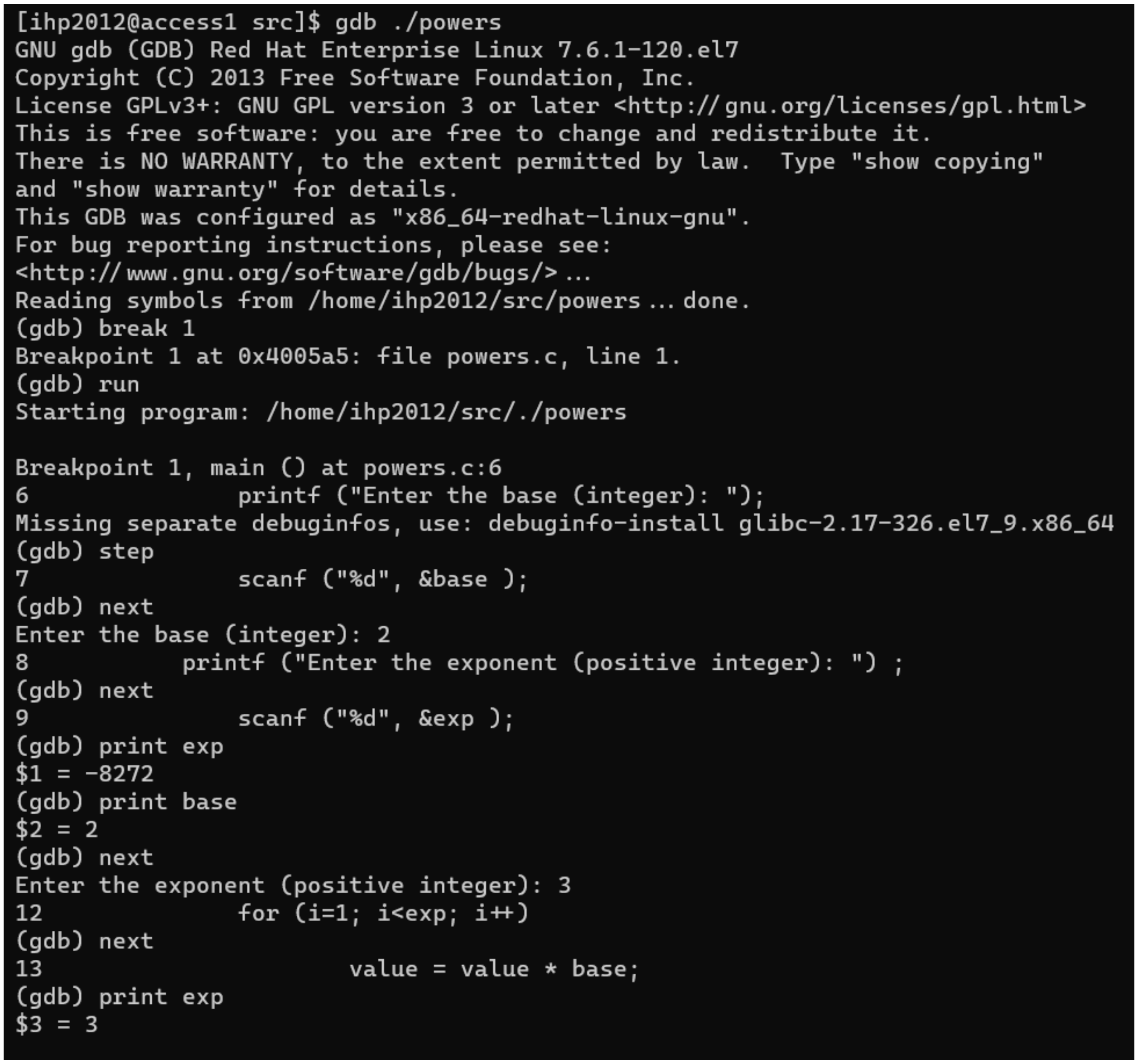
Exercise 1



```
[ihp2012@access1 src]$ gdb ./powers
GNU gdb (GDB) Red Hat Enterprise Linux 7.6.1-120.el7
Copyright (C) 2013 Free Software Foundation, Inc.
License GPLv3+: GNU GPL version 3 or later <http://gnu.org/licenses/gpl.html>
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.  Type "show copying"
and "show warranty" for details.
This GDB was configured as "x86_64-redhat-linux-gnu".
For bug reporting instructions, please see:
<http://www.gnu.org/software/gdb/bugs/>...
Reading symbols from /home/ihp2012/src/powers...done.
(gdb) break 1
Breakpoint 1 at 0x4005a5: file powers.c, line 1.
(gdb) run
Starting program: /home/ihp2012/src/./powers

Breakpoint 1, main () at powers.c:6
6               printf ("Enter the base (integer): ");
Missing separate debuginfos, use: debuginfo-install glibc-2.17-326.el7_9.x86_64
(gdb) step
7               scanf ("%d", &base );
(gdb) next
Enter the base (integer): 2
8           printf ("Enter the exponent (positive integer): ") ;
(gdb) next
9               scanf ("%d", &exp );
(gdb) print exp
$1 = -8272
(gdb) print base
$2 = 2
(gdb) next
Enter the exponent (positive integer): 3
12              for (i=1; i<exp; i++)
(gdb) next
13                      value = value * base;
(gdb) print exp
$3 = 3
```

Exercise 2

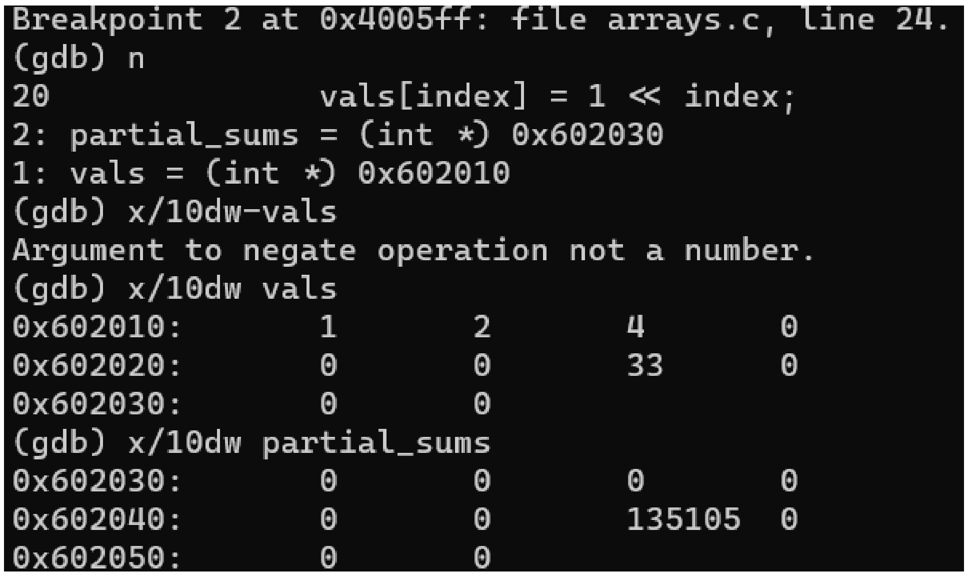
1. The memory address of the **vals** array is **0x602030**. The memory address of the **partial_sums** array is **0x602010.** I used the **display vals** and **display partial_sums** commands to discover these values.

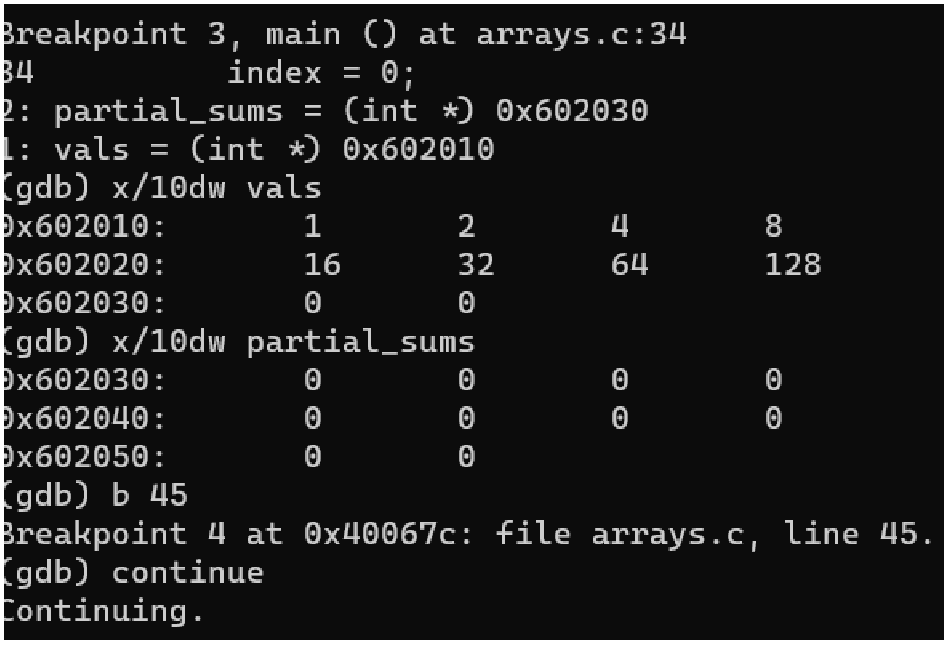
2. Once the loop between lines 18 and 22 completes, the values saved in the **vals** array are **1**, **2**, **4**, **0**, **0**, **0**, **33**, **0**, **0**, and **0**. I used the **x/10dw vals** command to discover these values.

3. Once the loop between lines 26 and 30 completes, the values saved in the **vals** array are **1**, **2**, **4**, **8**, **16**, **32**, **64**, **128**, **0**, and **0**. The values saved in the **partial_sums** array are all **0**. The numbers stored in the two arrays are not what they are supposed to be, since **vals[8]** and **vals[9]** should not be zero. However, **partial_sums** appears to have been zeroed-out correctly.

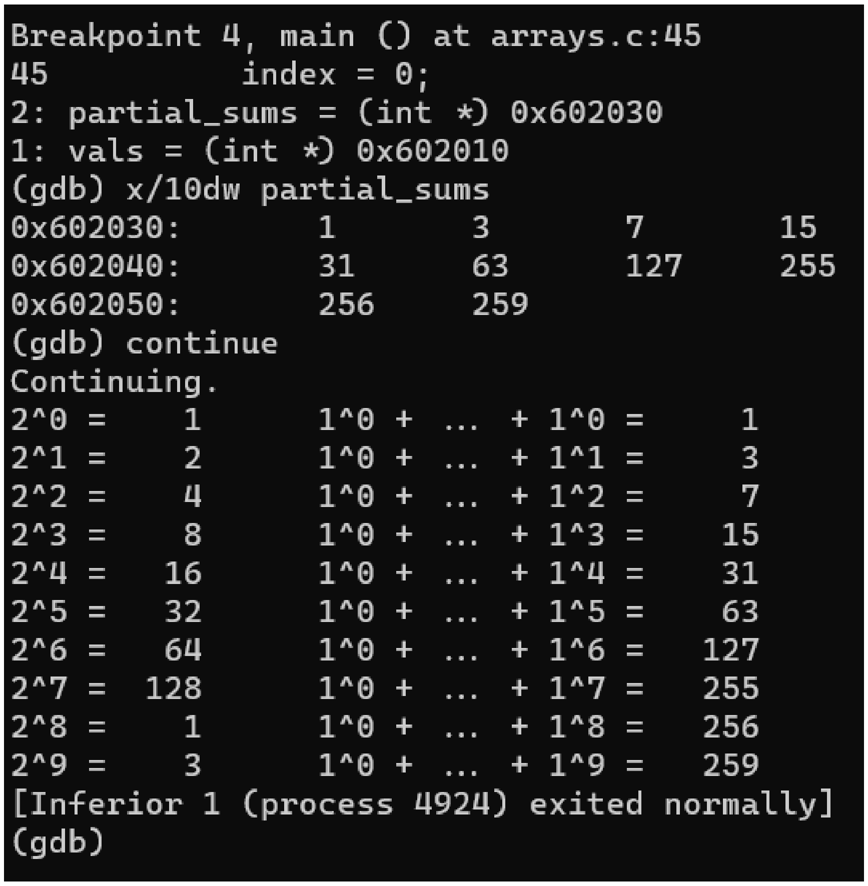
4. The difference between the two memory addresses in question 1 is **32**. This does not make sense because there are ten (32-bit integer) values in the first array, which means that at least 40 bytes worth of memory must be allocated. Since there is not enough room, 8 bytes (two integers) of the two arrays overlap.

5. Yes, the partial sums are printed correctly. However, the *last* two powers of two display are actually the *first two* partial sums.


```
Breakpoint 2 at 0x4005ff: file arrays.c, line 24.
(gdb) n
20              vals[index] = 1 << index;
2: partial_sums = (int *) 0x602030
1: vals = (int *) 0x602010
(gdb) x/10dw-vals
Argument to negate operation not a number.
(gdb) x/10dw vals
0x602010:       1       2       4       0
0x602020:       0       0       33      0
0x602030:       0       0
(gdb) x/10dw partial_sums
0x602030:       0       0       0       0
0x602040:       0       0       135105  0
0x602050:       0       0
```

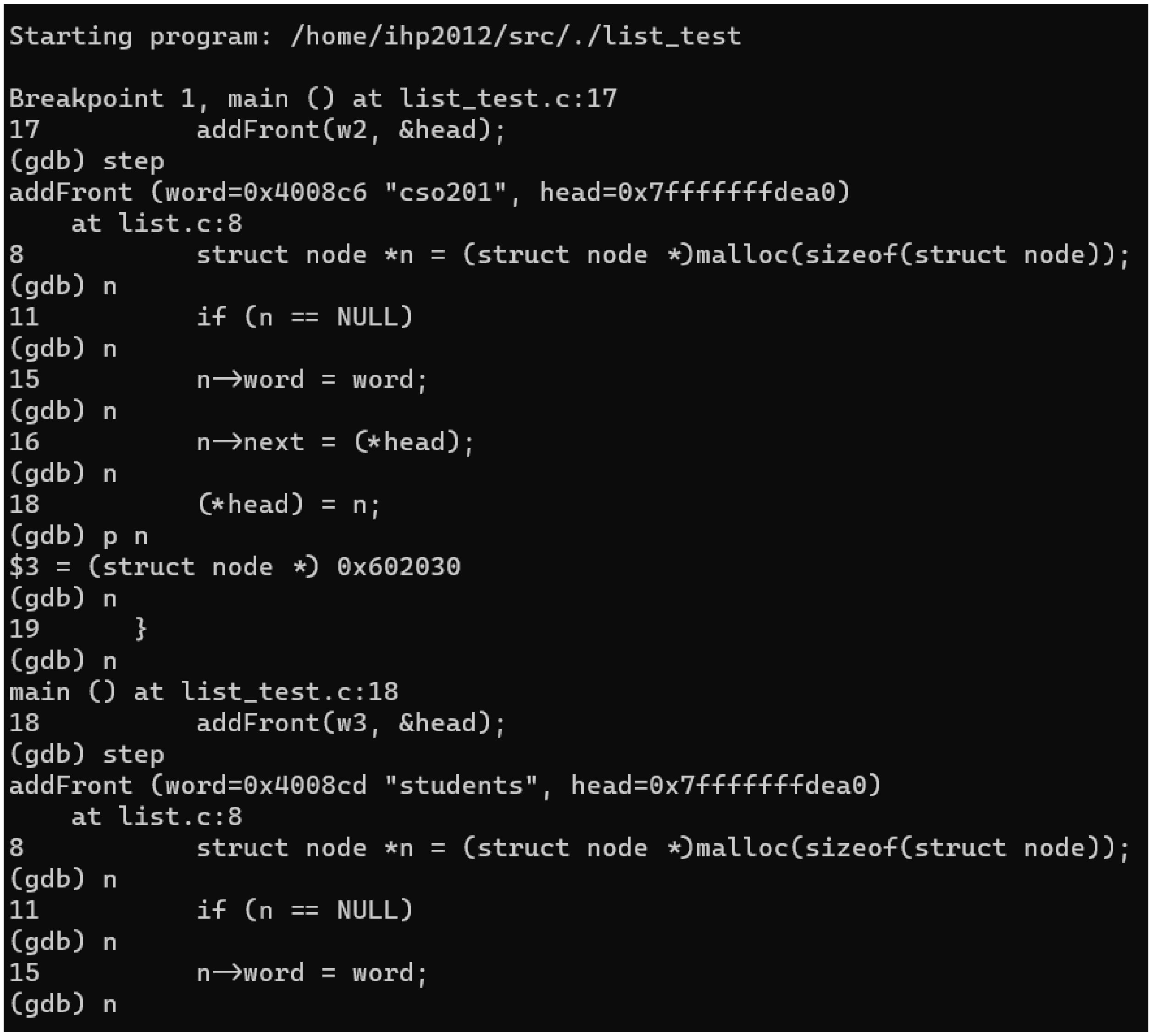


```
Breakpoint 3, main () at arrays.c:34
34          index = 0;
2: partial_sums = (int *) 0x602030
1: vals = (int *) 0x602010
(gdb) x/10dw vals
0x602010:       1       2       4       8
0x602020:       16      32      64      128
0x602030:       0       0
(gdb) x/10dw partial_sums
0x602030:       0       0       0       0
0x602040:       0       0       0       0
0x602050:       0       0
(gdb) b 45
Breakpoint 4 at 0x40067c: file arrays.c, line 45.
(gdb) continue
Continuing.
```

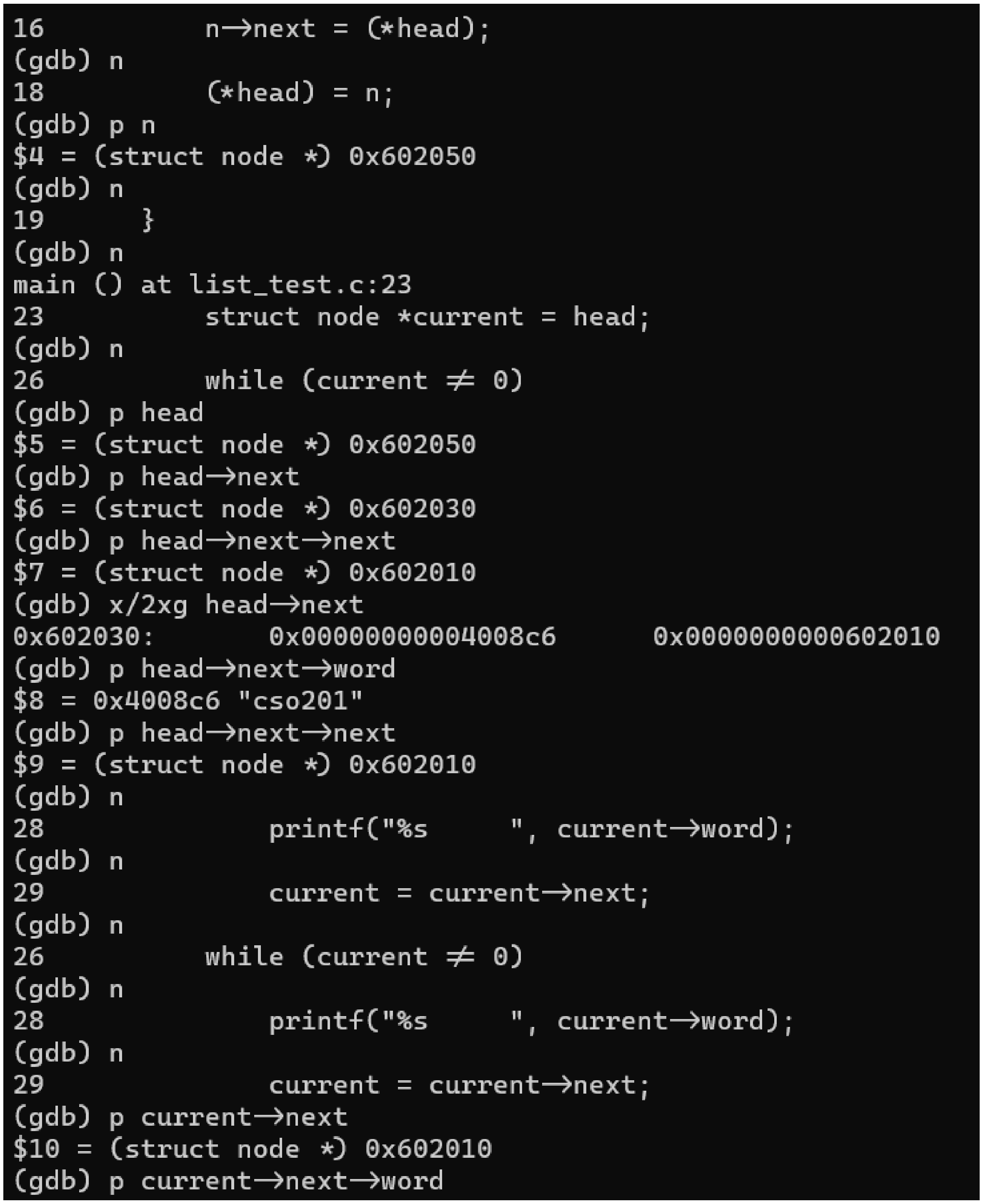


```
Breakpoint 4, main () at arrays.c:45
45          index = 0;
2: partial_sums = (int *) 0x602030
1: vals = (int *) 0x602010
(gdb) x/10dw partial_sums
0x602030:       1       3       7       15
0x602040:       31      63      127     255
0x602050:       256     259
(gdb) continue
Continuing.
2^0 =    1      1^0 +  ...  + 1^0 =     1
2^1 =    2      1^0 +  ...  + 1^1 =     3
2^2 =    4      1^0 +  ...  + 1^2 =     7
2^3 =    8      1^0 +  ...  + 1^3 =    15
2^4 =   16      1^0 +  ...  + 1^4 =    31
2^5 =   32      1^0 +  ...  + 1^5 =    63
2^6 =   64      1^0 +  ...  + 1^6 =   127
2^7 =  128      1^0 +  ...  + 1^7 =   255
2^8 =    1      1^0 +  ...  + 1^8 =   256
2^9 =    3      1^0 +  ...  + 1^9 =   259
[Inferior 1 (process 4924) exited normally]
(gdb)
```

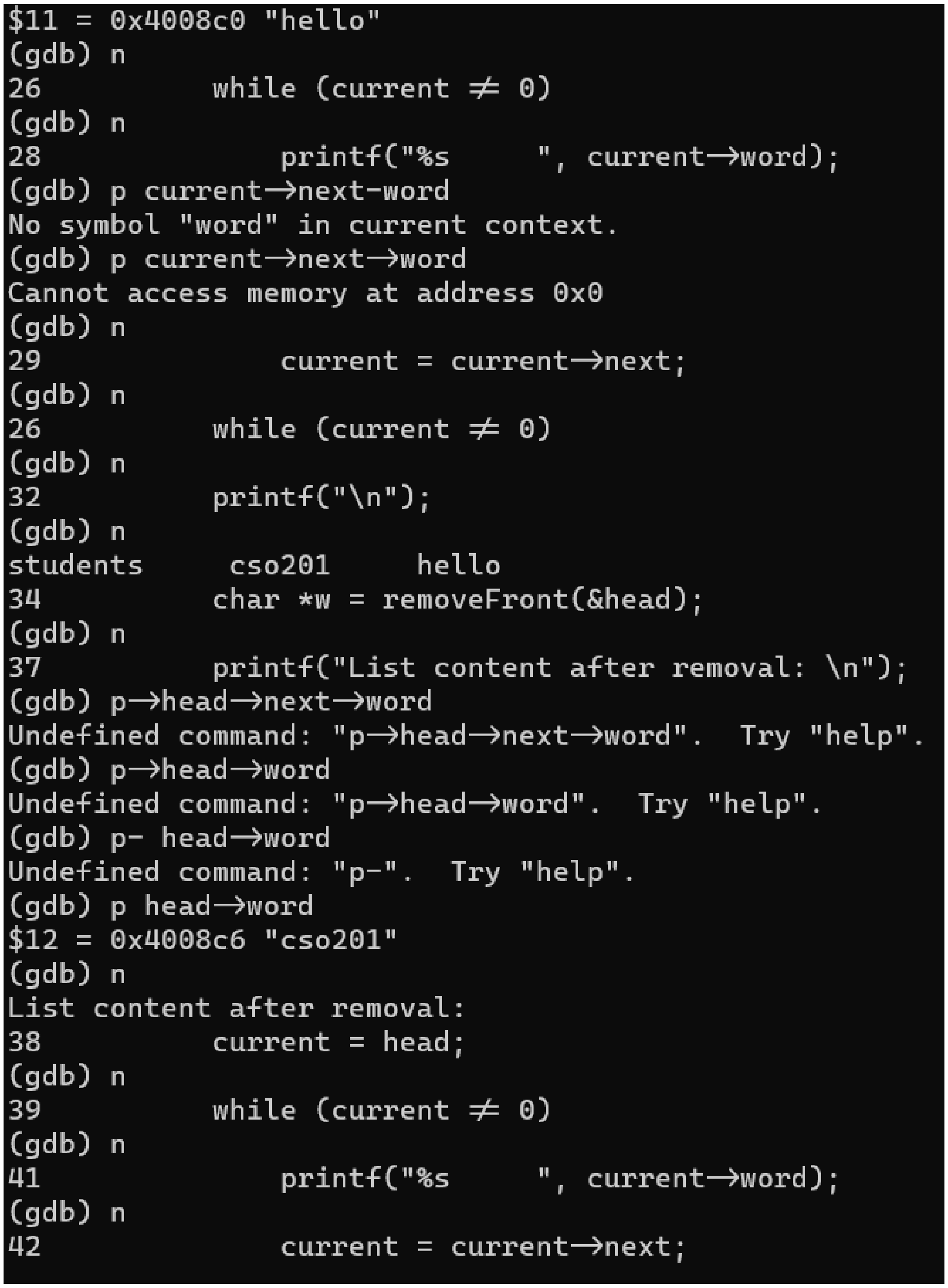
Exercise 3



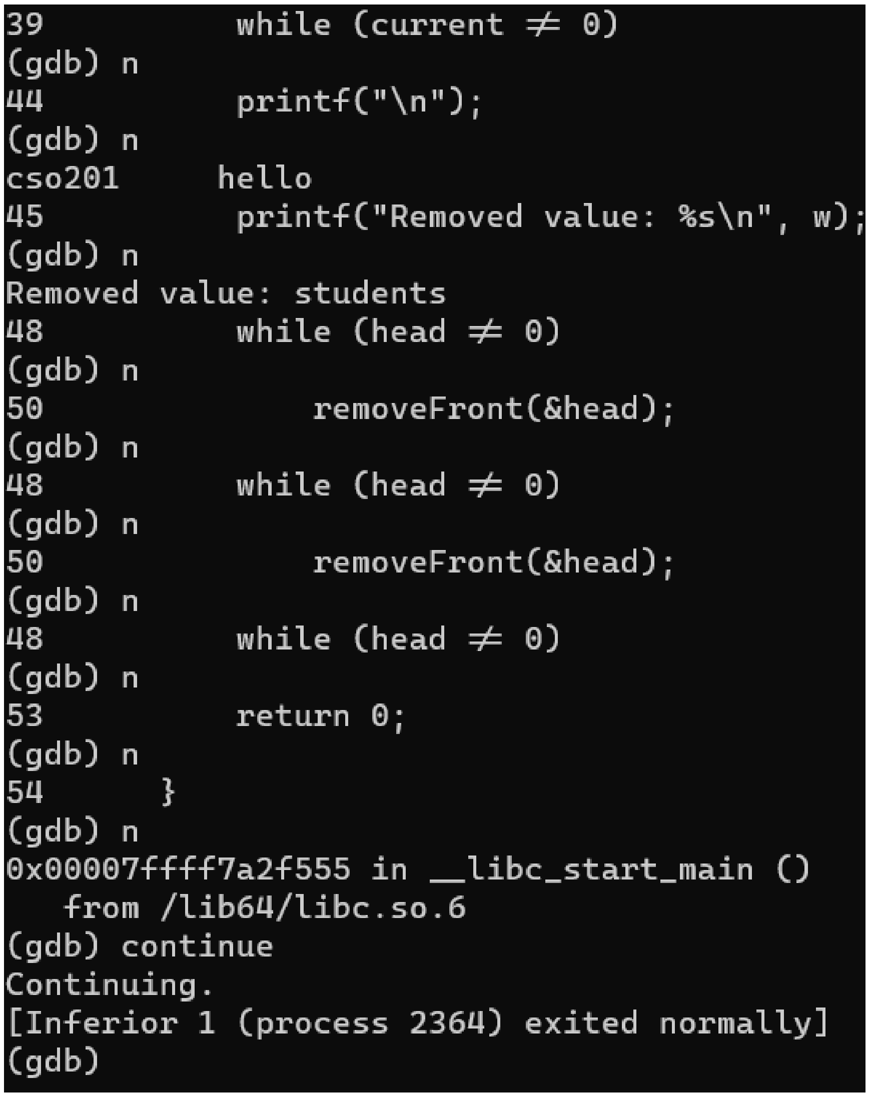
```
Starting program: /home/ihp2012/src/./list_test

Breakpoint 1, main () at list_test.c:17
17          addFront(w2, &head);
(gdb) step
addFront (word=0x4008c6 "cso201", head=0x7fffffffdea0)
    at list.c:8
8           struct node *n = (struct node *)malloc(sizeof(struct node));
(gdb) n
11          if (n == NULL)
(gdb) n
15          n->word = word;
(gdb) n
16          n->next = (*head);
(gdb) n
18          (*head) = n;
(gdb) p n
$3 = (struct node *) 0x602030
(gdb) n
19      }
(gdb) n
main () at list_test.c:18
18          addFront(w3, &head);
(gdb) step
addFront (word=0x4008cd "students", head=0x7fffffffdea0)
    at list.c:8
8           struct node *n = (struct node *)malloc(sizeof(struct node));
(gdb) n
11          if (n == NULL)
(gdb) n
15          n->word = word;
(gdb) n
```



```
16          n->next = (*head);
(gdb) n
18          (*head) = n;
(gdb) p n
$4 = (struct node *) 0x602050
(gdb) n
19      }
(gdb) n
main () at list_test.c:23
23          struct node *current = head;
(gdb) n
26          while (current != 0)
(gdb) p head
$5 = (struct node *) 0x602050
(gdb) p head->next
$6 = (struct node *) 0x602030
(gdb) p head->next->next
$7 = (struct node *) 0x602010
(gdb) x/2xg head->next
0x602030:       0x00000000004008c6      0x0000000000602010
(gdb) p head->next->word
$8 = 0x4008c6 "cso201"
(gdb) p head->next->next
$9 = (struct node *) 0x602010
(gdb) n
28              printf("%s     ", current->word);
(gdb) n
29              current = current->next;
(gdb) n
26          while (current != 0)
(gdb) n
28              printf("%s     ", current->word);
(gdb) n
29              current = current->next;
(gdb) p current->next
$10 = (struct node *) 0x602010
(gdb) p current->next->word
```



```
$11 = 0x4008c0 "hello"
(gdb) n
26          while (current ≠ 0)
(gdb) n
28              printf("%s     ", current→word);
(gdb) p current→next-word
No symbol "word" in current context.
(gdb) p current→next→word
Cannot access memory at address 0x0
(gdb) n
29              current = current→next;
(gdb) n
26          while (current ≠ 0)
(gdb) n
32          printf("\n");
(gdb) n
students     cso201     hello
34          char *w = removeFront(&head);
(gdb) n
37          printf("List content after removal: \n");
(gdb) p→head→next→word
Undefined command: "p→head→next→word".  Try "help".
(gdb) p→head→word
Undefined command: "p→head→word".  Try "help".
(gdb) p- head→word
Undefined command: "p-".  Try "help".
(gdb) p head→word
$12 = 0x4008c6 "cso201"
(gdb) n
List content after removal:
38          current = head;
(gdb) n
39          while (current ≠ 0)
(gdb) n
41              printf("%s     ", current→word);
(gdb) n
42              current = current→next;
```



```
39          while (current ≠ 0)
(gdb) n
44          printf("\n");
(gdb) n
cso201     hello
45          printf("Removed value: %s\n", w);
(gdb) n
Removed value: students
48          while (head ≠ 0)
(gdb) n
50              removeFront(&head);
(gdb) n
48          while (head ≠ 0)
(gdb) n
50              removeFront(&head);
(gdb) n
48          while (head ≠ 0)
(gdb) n
53          return 0;
(gdb) n
54      }
(gdb) n
0x00007ffff7a2f555 in __libc_start_main ()
   from /lib64/libc.so.6
(gdb) continue
Continuing.
[Inferior 1 (process 2364) exited normally]
(gdb)
```