memset showing wrong results [duplicate]

This question already has an answer here:
Initializing entire array with memset *4 answers*

The code is to check the working of memset in C. memset initializes the array correctly for 0 but when I try initialize it with 10 it initializes the array with some very large garbage value .What's wrong?

```
#include <stdio.h>
#include<string.h>
int main(void)
{
    int dp[10008],i;
    memset(dp,10,sizeof(dp));
    for(i=0;i<10;i++)
    printf("%d\n",dp[i]);
    return 0;
}
```

c memset

edited Sep 29 '15 at 11:42
utnapistim
20.7k 2 27 70

asked Sep 29 '15 at 11:29
Jatin Narula
1 2

marked as duplicate by Jens Gustedt c Sep 29 '15 at 13:26

This question has been asked before and already has an answer. If those answers do not fully address your question, please ask a new question.

6 memset initialises each byte with 10. when you interpret four bytes with 10 in them (or 8 or whatever) then you get these values.. you need to loop over the data and set each integer (not byte) to 10 if thats what you want – amdixon Sep 29 '15 at 11:32

2 @amdixon: Yep. Why not write this as an answer? – M Oehm Sep 29 '15 at 11:32

@MOehm ok will do – amdixon Sep 29 '15 at 11:33

2 While *a* solution to this could be the same in both C and C++, C++ have other alternatives that would be better (IMO). Please pick the language you want the solution in. – Some programmer dude Sep 29 '15 at 11:34

2 You shouldn't allocate large amounts of data on the stack. That is, unless you are a fan of stack overflow. – Lundin Sep 29 '15 at 11:39

2 Answers

man memset

> ```
>        void *memset(void *s, int c, size_t n);
>
> DESCRIPTION
>        The  memset()  function  fills  the  first  n  bytes of the memory area
>        pointed to by s with the constant byte c.
> ```

your code :

```
memset(dp,10,sizeof(dp));
```

then is initialising all bytes in `dp` to 10.

so your array looks like (bytewise) :

```
+----------+----------+----------+----------+
| 00001010 | 00001010 | 00001010 | 00001010 | ...
+----------+----------+----------+----------+
```

and if you interpret that as an integer you get (some large value).

note above is based on sizeof int as 4 for illustrative purposes only

```c
#include <stdio.h>
#include <string.h>
int main(void)
{
    int dp[10008],i;
    for(i=0;i<10008;i++)
    {
      dp[i] = 10;
    }
    // do other stuff here..
    return 0;
}
```

edited Sep 29 '15 at 12:34

answered Sep 29 '15 at 11:40

amdixon

3,496 8 18 32

The memset should be removed entirely from the modified code. – Lundin Sep 29 '15 at 11:41

thanks, too much copy+paste ;) – amdixon Sep 29 '15 at 11:42

4 In case anybody is tempted to apply premature optimization to your good, clear code: compilers are pretty good at optimizing simple loops and I'd expect that the compiler would convert this loop to a big memset anyway – M.M Sep 29 '15 at 11:45

@LPs `memset` sets bytes to `value` . Every byte is 8 bits, and 10 is `0x00001010` , so why would it be `0x10101010` ? – Enzo Ferber Sep 29 '15 at 12:13

@EnzoFerber I misunderstood. I thought was the array representation, not the byte binary representation. – LPs Sep 29 '15 at 12:32

memset sets bytes and works for characters because they are single bytes, but integers are not.

answered Sep 29 '15 at 11:44

Vinod

253 2 14