CS 214 Recitation (Sec.7)

Sept. 26, 2017

Topic of this week

GDB - GNU Project Debugger

Dynamic Memory Allocation

Structs Sizeof

String Library

GDB - Debugging in Linux

- gdb can only work with executable file (not .c .h)
- need to use argument -g when compile: gcc -g test.c -o test
- running the code: using run (r) or continue (c)
- using list (l) to see 10 lines of code (or list n)

GDB - Debugging in Linux

- setting break points: using break n (b n) or break func
- clearing breakpoints: using delete n (delete breakpoints) or disable n (enable to reset)
- looking at variables: using print name (or p name) or display a or info locals
- stepping through code: using next(n) - not enter funcs or step(s) - enter funcs
- quit gdb: quit (q)

Dynamic Memory Allocation

- malloc

  void malloc (size_t size);* Assign a fixed size of memory, but not initialize it.

- calloc

  void calloc (size_t n, size_t s);* Assign and initial n successive memory spaces of size s to 0

Dynamic Memory Allocation

- realloc

  void realloc (void* ptr, size_t s);* For the memory which ptr points to, assign a memory space of size s

- free

  *char *ptr = (char*) calloc (10,10); free(ptr);*

```c
#include <stdio.h>
#include <stdlib.h>

int main()
{
   char *str;

   /* Initial memory allocation */
   str = (char *) malloc(20);
   printf("String = %s,  Address = %u\n", str, str);
   strcpy(str, "cs214_recitation");
   printf("String = %s,  Address = %u\n", str, str);

   /* Reallocating memory */
   str = (char *) realloc(str, 30);
   strcat(str, ".section7");
   printf("String = %s,  Address = %u\n", str, str);


   free(str);

   return(0);
}
```

```
~/2017F/CS 214/Recitation » ./alloc
String = ,  Address = 1514153536
String = cs214_recitation,  Address = 1514153536
String = cs214_recitation.section7,  Address = 1514153536
```

Structs Sizeof

```
struct MyStruct
{
  double dda1;
  char dda;
  int type;
};
```

What is the value of sizeof(MyStruct)?

Is it 8+1+4 ?

```
Union MyUnion
{
  double dda1;
  char dda;
  int type;
};
```

What is the value of sizeof(MyUnion)?

Structs Sizeof

```
struct MyStruct
{
  double dda1;
  char dda;
  int type;
};
```

What is the value of sizeof(MyStruct)?

Is it 8+1+4 ? No

Alignment - to speed up the memory manipulation

double 8 (* * * * * * * *)

char 1 (* _ _ _)

int 4 (* * * *)

Is it 8+1+4 ? No

It is 8+1+3+4 = 16 = 8 * 2

No need to pad

Structs Sizeof

```
struct MyStruct
{
  char dda;
  double dda1;
  int type;
};
```

What is the value of sizeof(MyStruct)?

Is it 1+8+4 ? No

Alignment - to speed up the memory manipulation

char 1 (*_ _ _ _ _ _ _)

double 8 (* * * * * * * *)

int 4 (* * * *_ _ _ _)

Need to pad

Is it 1+7+8+4 ? No

It is 1+7+8+4 + (4) = 24= 8 * 3

String Library

- Initialize a string:

char greeting[] = "Hello";

*char *ptr = (char*) malloc (sizeof(char)*10);*

- #include <string.h>

  strcpy strlen strcat memset memcpy strtok

String Library

- strcpy strlen strcat

```
strcpy( str3, str1) :  Hello
strcat( str1, str2):   HelloWorld
strlen(str1) :  10
```

```c
#include <stdio.h>
#include <string.h>

int main () {

   char str1[12] = "Hello";
   char str2[12] = "World";
   char str3[12];
   int  len ;

   /* copy str1 into str3 */
   strcpy(str3, str1);
   printf("strcpy( str3, str1) :  %s\n", str3 );

   /* concatenates str1 and str2 */
   strcat( str1, str2);
   printf("strcat( str1, str2):   %s\n", str1 );

   /* total lenghth of str1 after concatenation */
   len = strlen(str1);
   printf("strlen(str1) :  %d\n", len );

   return 0;
}
```

String Library

- memset memcpy

*void * memset(void * ptr, int v, size_t n);* n means the first n bits

*void * memcpy (void * dest, const void * src, size_t n);*

String Library

- memset

-------hijkmln

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>

int main()
{
        char str[] = "abcdefghijkmln";
        memset(str, '-', 7);
        puts(str);
        return 0;
}
```

String Library

- memcpy

```
p2 = abcde
p3 = abcde
```

```c
#include <stdio.h>
#include <stdlib.h>
#include <string.h>

#define N (10)

int main()
{
    char* p1 = "abcde";
    char* p2 = (char*)malloc(sizeof(char) * N);
    char* p3 = (char*)memcpy(p2, p1, N);
    printf("p2 = %s\np3 = %s\n", p2, p3);

    free(p2);
    p2 = NULL;
    p3 = NULL;
    system("pause");
    return 0;
}
```

String Library

- strtok

*char * strtok(char *s, const char *delim);*

ab cd ef;gh i jkl;mnop;qrs tu vwx y;z

```
#include <string.h>
main(){
    char s[] = "ab-cd : ef;gh :i-jkl;mnop;qrs-tu: vwx-y;z";
    char *delim = "-: ";
    char *p;
    printf("%s ", strtok(s, delim));
    while((p = strtok(NULL, delim)))
        printf("%s ", p);
        printf("\n");
}
```

Homework 1 - Q0&Q1 Answer

0. What's wrong with this #define line?

    #define N 10;

1. Suppose you defined the macro

    #define SIX 2*3

    Then, suppose you used it in another expression:

    int x = 12 / SIX; int x = 12 / 2*3;

    What value would x be set to?

How about #define SIX (2*3) ?

Homework 1 - Q2 Answer

2. Write your own version of atoi

Take a char, inspect its int value and return its corresponding int value

e.g.

```
int test = my_atoi('5');
if( test == 5 )
{
        return 0;
}
else
{
        return -1;
}
```

Next, take a string of any length, scan its chars until you hit the '\0' and return the entire string's int value

e.g.

```
int test = my_atoi("512");
if( test == 512 )
{
        return 0;
}
else
{
        return -1;
}
```

Homework 1 - Q2 Answer

What will atoi() return?

if char *p = “-15a”; output = atoi(p); output = -15

if char *p=“+129akd323”; output = 129

if char *p=“ 1290”; output = 1290

if char *p=“*11asdf1322”; output = 0

if char *p=“a567q”; output = 0

But atoi() in the <stdlib.h> has a problem with handling overflow!

Homework 1 - Q2 Answer

Some key points to notice:

1. integers consist of positive int & negative int (‘+’ & ‘-’)

2. take overflow into account

3. how to deal with string that starts with one or more spaces

4. how to deal with characters that are not ‘0’-’9’

A LeetCode Medium level question (No.8) *13.9%* Acceptance

```c
#include <stdio.h>
#include <stdlib.h>

// value range of int type
#define MAX_INT ((1 << 31) - 1)  //  2147483647
#define MIN_INT (-(1 << 31))     //  -2147483648

int my_atoi(const char *str);

int main()
{
    char *p;
    scanf("%s",p);
    printf("myAtoi:%d\n", my_atoi(p));
    printf("atoi:%d\n", atoi(p));
    return 0;
}
```

```c
int my_atoi(const char *str)
{
    long long value = 0;  // use to calculate the integer value of user input
    int flag = 1; //positive flag

    while (*str == ' ')  //skip the spaces
    {
        str++;
    }

    if (*str == '-')  //if the first char is '-' may be a neg num
    {
        flag = 0;
        str++;
    }
    else if (*str == '+') //if the first char is '+' may be a pos num
    {
        flag = 1;
        str++;
    }//if the first char is not '+' or '-' or num char, return 0
    else if (*str >= '9' || *str <= '0')
    {
        return 0;
    }
```

```
//when next char is not num char or encounters '/0', transfer ends
while (*str != '/0' && *str <= '9' && *str >= '0')
{
    value = value * 10 + (*str - '0'); //trasfer num char to integer
    //printf("%1.12lld\n",value);
    str++;
    if (flag && value>MAX_INT)
    {
        return MAX_INT;
    }
    if (!flag && -value<MIN_INT)
    {

        return MIN_INT;
    }

}

if (flag == 0) //for neg num, and a neg sign
{
    value = -value;
}

return value;
```

ASCII (1977/1986)

	_0	_1	_2	_3	_4	_5	_6	_7	_8	_9	_A	_B	_C	_D	_E	_F
0_	NUL 0000 *0*	SOH 0001 *1*	STX 0002 *2*	ETX 0003 *3*	EOT 0004 *4*	ENQ 0005 *5*	ACK 0006 *6*	BEL 0007 *7*	BS 0008 *8*	HT 0009 *9*	LF 000A *10*	VT 000B *11*	FF 000C *12*	CR 000D *13*	SO 000E *14*	SI 000F *15*
1_	DLE 0010 *16*	DC1 0011 *17*	DC2 0012 *18*	DC3 0013 *19*	DC4 0014 *20*	NAK 0015 *21*	SYN 0016 *22*	ETB 0017 *23*	CAN 0018 *24*	EM 0019 *25*	SUB 001A *26*	ESC 001B *27*	FS 001C *28*	GS 001D *29*	RS 001E *30*	US 001F *31*
2_	SP 0020 *32*	! 0021 *33*	" 0022 *34*	# 0023 *35*	$ 0024 *36*	% 0025 *37*	& 0026 *38*	' 0027 *39*	(0028 *40*	) 0029 *41*	* 002A *42*	+ 002B *43*	, 002C *44*	- 002D *45*	. 002E *46*	/ 002F *47*
3_	0 0030 *48*	1 0031 *49*	2 0032 *50*	3 0033 *51*	4 0034 *52*	5 0035 *53*	6 0036 *54*	7 0037 *55*	8 0038 *56*	9 0039 *57*	: 003A *58*	; 003B *59*	< 003C *60*	= 003D *61*	> 003E *62*	? 003F *63*
4_	@ 0040 *64*	A 0041 *65*	B 0042 *66*	C 0043 *67*	D 0044 *68*	E 0045 *69*	F 0046 *70*	G 0047 *71*	H 0048 *72*	I 0049 *73*	J 004A *74*	K 004B *75*	L 004C *76*	M 004D *77*	N 004E *78*	O 004F *79*
5_	P 0050 *80*	Q 0051 *81*	R 0052 *82*	S 0053 *83*	T 0054 *84*	U 0055 *85*	V 0056 *86*	W 0057 *87*	X 0058 *88*	Y 0059 *89*	Z 005A *90*	[005B *91*	\ 005C *92*	] 005D *93*	^ 005E *94*	_ 005F *95*
6_	` 0060 *96*	a 0061 *97*	b 0062 *98*	c 0063 *99*	d 0064 *100*	e 0065 *101*	f 0066 *102*	g 0067 *103*	h 0068 *104*	i 0069 *105*	j 006A *106*	k 006B *107*	l 006C *108*	m 006D *109*	n 006E *110*	o 006F *111*
7_	p 0070 *112*	q 0071 *113*	r 0072 *114*	s 0073 *115*	t 0074 *116*	u 0075 *117*	v 0076 *118*	w 0077 *119*	x 0078 *120*	y 0079 *121*	z 007A *122*	{ 007B *123*	| 007C *124*	} 007D *125*	~ 007E *126*	DEL 007F *127*

Homework 1 - Q2 Answer

Running Examples:

```
-45635655474563sdfa!!@R%%%@#%^
000000000004
000000000045
000000000456
000000004563
000000045635
000000456356
000004563565
000045635655
000456356554
004563565547
myAtoi:-2147483648
atoi:-1627954563
```

```
24232423434asdfjialef%%^#W#$%Asdf3222
000000000002
000000000024
000000000242
000000002423
000000024232
000000242324
000002423242
000024232423
000242324234
002423242343
myAtoi:2147483647
atoi:-1537380342
```

Homework 1 - Q2 Answer

Running Examples:

```
    134a4555
000000000001
000000000013
000000000134
myAtoi:134
atoi:134
```

```
    a$#$$2344
myAtoi:0
atoi:0
```

Homework 2

0. Consider:

```
int i = 5;

int *ip = &i;
```

.. what is ip? What is its value?

Homework 2 (Cont.)

1. Write some code that declares two arrays of size 10 that are string literals.

    Make a pointer to one of the arrays, cast it to be an int pointer, and print out its value.

    Make a new integer, set it equal to the value of your int pointer, then make a pointer to that integer, cast it to be a char pointer, and print out 8 chars.

    What happened? Why?

Homework 2 (Cont.)

2. Write some code that declares two arrays of size 10 that are string literals.

Create a pointer that points to the beginning of the first array, then in a loop, increment the pointer and print out the char it points to, out to index 20.

What happened? Why?