[CS200]-Software Tools and Technologies Lab-II **Fall 2020-21**

Homework: IV

Team Name: Peaky Blinders RollNo:11940150,11940370,11940380

2a]

The code given in the question was saved in a file named **hw42a.c**.
After that the following commands were run in the terminal :-
1] **gcc -g hw42a.c**
The above command will compile the hw42a.c file and gcc -g generates debug information to be used by the debugger.
2]**valgrind –leak-check=yes ./a.out**
Memcheck is the default tool. The –leak-check option turns on the detailed memory leak detector.
The output of 2nd command is given below :-

```
ritik@ritik-HP-Pavilion-Laptop-15-cs3xxx:~$ gcc -g hw42a.c
ritik@ritik-HP-Pavilion-Laptop-15-cs3xxx:~$ valgrind --leak-check=yes ./a.out
==23604== Memcheck, a memory error detector
==23604== Copyright (C) 2002-2017, and GNU GPL'd, by Julian Seward et al.
==23604== Using Valgrind-3.15.0 and LibVEX; rerun with -h for copyright info
==23604== Command: ./a.out
==23604==
iArray[516]: 516
==23604==
==23604== HEAP SUMMARY:
==23604==     in use at exit: 4,000 bytes in 1 blocks
==23604==   total heap usage: 2 allocs, 1 frees, 5,024 bytes allocated
==23604==
==23604== 4,000 bytes in 1 blocks are definitely lost in loss record 1 of 1
==23604==    at 0x483B7F3: malloc (in /usr/lib/x86_64-linux-gnu/valgrind/vgpreload_memcheck-amd64-linux.so)
==23604==    by 0x1091E7: main (hw42a.c:6)
==23604==
==23604== LEAK SUMMARY:
==23604==    definitely lost: 4,000 bytes in 1 blocks
==23604==    indirectly lost: 0 bytes in 0 blocks
==23604==      possibly lost: 0 bytes in 0 blocks
==23604==    still reachable: 0 bytes in 0 blocks
==23604==         suppressed: 0 bytes in 0 blocks
==23604==
==23604== For lists of detected and suppressed errors, rerun with: -s
==23604== ERROR SUMMARY: 1 errors from 1 contexts (suppressed: 0 from 0)
ritik@ritik-HP-Pavilion-Laptop-15-cs3xxx:~$
```

By seeing the output we know that a **memory leak** of type **definitely lost** has occured.
Final error message given in the output is :-
1 error from 1 contexts.
Now the error needs to be rectified.
The main reason for the error is that we have **allocated** the memory but we have not **deallocated** it.

We have to use the **free() function**.
I used the free() function in the given code to correct the error

The code with correction is given below :-

```c
#include<stdlib.h>
#include<stdio.h>
#include<time.h>
const int SIZE = 1000;
int main() {
int *iArray = malloc(sizeof(int) * SIZE);
for (int i=0; i < SIZE; i++) {
iArray[i] = i;
}
srand(time(NULL));
int randNum = rand() % SIZE;
printf("iArray[%d]: %d\n", randNum, iArray[randNum]);
free(iArray);           // free() function is used to de-allocate the memory allocated by the
                        // malloc() at line no. 6
return 0;
}
```

After this change I saved the file **hw42a.c** again and ran the following commands :-
1] **gcc -g hw42a.c**
2]**valgrind –leak-check=yes ./a.out**

The output of the 2nd command is given below :-

```
ritik@ritik-HP-Pavilion-Laptop-15-cs3xxx:~$ gcc -g hw42a.c
ritik@ritik-HP-Pavilion-Laptop-15-cs3xxx:~$ valgrind --leak-check=yes ./a.out
==6987== Memcheck, a memory error detector
==6987== Copyright (C) 2002-2017, and GNU GPL'd, by Julian Seward et al.
==6987== Using Valgrind-3.15.0 and LibVEX; rerun with -h for copyright info
==6987== Command: ./a.out
==6987==
iArray[426]: 426
==6987==
==6987== HEAP SUMMARY:
==6987==     in use at exit: 0 bytes in 0 blocks
==6987==   total heap usage: 2 allocs, 2 frees, 5,024 bytes allocated
==6987==
==6987== All heap blocks were freed -- no leaks are possible
==6987==
==6987== For lists of detected and suppressed errors, rerun with: -s
==6987== ERROR SUMMARY: 0 errors from 0 contexts (suppressed: 0 from 0)
ritik@ritik-HP-Pavilion-Laptop-15-cs3xxx:~$
```

As you can see in the output ERROR SUMMARY is :-
0 error from 0 contexts.

2b]

The code given in the question was saved in a file named **hw42b.c**.
After that the following commands were run in the terminal :-
1] **gcc -g hw42b.c**
The above command will compile the hw42b.c file and gcc -g generates debug information to be used by the debugger.
2]**valgrind –leak-check=yes ./a.out**
Memcheck is the default tool. The –leak-check option turns on the detailed memory leak detector.
The output of 2nd command is given below :-

```
ritik@ritik-HP-Pavilion-Laptop-15-cs3xxx:~$ gcc -g hw42b.c
ritik@ritik-HP-Pavilion-Laptop-15-cs3xxx:~$ valgrind --leak-check=yes ./a.out
==10307== Memcheck, a memory error detector
==10307== Copyright (C) 2002-2017, and GNU GPL'd, by Julian Seward et al.
==10307== Using Valgrind-3.15.0 and LibVEX; rerun with -h for copyright info
==10307== Command: ./a.out
==10307==
==10307== Invalid free() / delete / delete[] / realloc()
==10307==    at 0x483CA3F: free (in /usr/lib/x86_64-linux-gnu/valgrind/vgpreload_memcheck-amd64-linux.so)
==10307==    by 0x109215: main (hw42b.c:17)
==10307==  Address 0x4a52090 is 0 bytes inside a block of size 40 free'd
==10307==    at 0x483DFAF: realloc (in /usr/lib/x86_64-linux-gnu/valgrind/vgpreload_memcheck-amd64-linux.so)
==10307==    by 0x1091B4: resizeArray (hw42b.c:10)
==10307==    by 0x109202: main (hw42b.c:16)
==10307==  Block was alloc'd at
==10307==    at 0x483DD99: calloc (in /usr/lib/x86_64-linux-gnu/valgrind/vgpreload_memcheck-amd64-linux.so)
==10307==    by 0x1091EC: main (hw42b.c:15)
==10307==
==10307==
==10307== HEAP SUMMARY:
==10307==     in use at exit: 60 bytes in 1 blocks
==10307==   total heap usage: 3 allocs, 3 frees, 116 bytes allocated
==10307==
==10307== 60 bytes in 1 blocks are definitely lost in loss record 1 of 1
==10307==    at 0x483DFAF: realloc (in /usr/lib/x86_64-linux-gnu/valgrind/vgpreload_memcheck-amd64-linux.so)
==10307==    by 0x1091B4: resizeArray (hw42b.c:10)
==10307==    by 0x109202: main (hw42b.c:16)
==10307==
==10307== LEAK SUMMARY:
==10307==    definitely lost: 60 bytes in 1 blocks
==10307==    indirectly lost: 0 bytes in 0 blocks
==10307==      possibly lost: 0 bytes in 0 blocks
==10307==    still reachable: 0 bytes in 0 blocks
==10307==         suppressed: 0 bytes in 0 blocks
==10307==
==10307== For lists of detected and suppressed errors, rerun with: -s
==10307== ERROR SUMMARY: 2 errors from 2 contexts (suppressed: 0 from 0)
```

Final error message given in the output is :-
2 error from 2 contexts.
There are two different kinds of error and this we can tell by looking at the output message. The two errors are given below :-
1] **Invalid free()**
2] **Memory Leak** of **definitely lost** type.
Let’s resolve the errors one by one :-

The **Invalid free** errors occurs when we try to free invalid memory (or free same memory more than once).
Let’s look at the given code :-

```
 1 #include <stdlib.h>
 2 #include <stdint.h>
 3 struct _List {
 4 int32_t* data;
 5 int32_t length;
 6 };
 7 typedef struct _List List;
 8 List* resizeArray(List* array) {
 9 int32_t* dPtr = array->data;
10 dPtr = realloc(dPtr, 15 * sizeof(int32_t));
11 return array;
12 }
13 int main() {
14 List* array = calloc(1, sizeof(List));
15 array->data = calloc(10, sizeof(int32_t));
16 array = resizeArray(array);
17 free(array->data);
18 free(array);
19 return 0;
20 }
```

The invalid free error is given in the line no. 17 i.e. **free(array− >data)**.
And here we are trying to free an invalid memory. We can free(array) which is done in line no. 18.
So to resolve this error we need to remove line no. 17 as the work of **free(array− >data)** is done by **free(array)**.
I removed the line no. 17 from the code and then saved the file **hw42b.c**.
After that I ran the following command again :-
1] **gcc -g hw42b.c**
2]**valgrind –leak-check=yes ./a.out**
The output is given below :-

```
ritik@ritik-HP-Pavilion-Laptop-15-cs3xxx:~$ gcc -g hw42b.c
ritik@ritik-HP-Pavilion-Laptop-15-cs3xxx:~$ valgrind --leak-check=yes ./a.out
==10448== Memcheck, a memory error detector
==10448== Copyright (C) 2002-2017, and GNU GPL'd, by Julian Seward et al.
==10448== Using Valgrind-3.15.0 and LibVEX; rerun with -h for copyright info
==10448== Command: ./a.out
==10448==
==10448==
==10448== HEAP SUMMARY:
==10448==     in use at exit: 60 bytes in 1 blocks
==10448==   total heap usage: 3 allocs, 2 frees, 116 bytes allocated
==10448==
==10448== 60 bytes in 1 blocks are definitely lost in loss record 1 of 1
==10448==    at 0x483DFAF: realloc (in /usr/lib/x86_64-linux-gnu/valgrind/vgpreload_memcheck-amd64-linux.so)
==10448==    by 0x1091B4: resizeArray (hw42b.c:10)
==10448==    by 0x109202: main (hw42b.c:16)
==10448==
==10448== LEAK SUMMARY:
==10448==    definitely lost: 60 bytes in 1 blocks
==10448==    indirectly lost: 0 bytes in 0 blocks
==10448==      possibly lost: 0 bytes in 0 blocks
==10448==    still reachable: 0 bytes in 0 blocks
==10448==         suppressed: 0 bytes in 0 blocks
==10448==
==10448== For lists of detected and suppressed errors, rerun with: -s
==10448== ERROR SUMMARY: 1 errors from 1 contexts (suppressed: 0 from 0)
ritik@ritik-HP-Pavilion-Laptop-15-cs3xxx:~$
```

As we can see that only one error is left and the invalid free error has been solved.

Now the next error is memory leak of definitely lost type. If we look at the output message it states that error is at **line no. 10 and 16** of the code.
By this we can tell that the 2nd error is in the function **resizeArray** of the code.
dPtr is allocated memory but it hasn't been deallocated and that's why the error occurred.

free(dPtr); should be added in the function **resizeArray** before the line **return(array)**.
The final code after all the modifications is given below :-

```c
#include <stdlib.h>
#include <stdint.h>
struct _List {
int32_t* data;
int32_t length;
};
typedef struct _List List;
List* resizeArray(List* array) {
int32_t* dPtr = array->data;
dPtr = realloc(dPtr, 15 * sizeof(int32_t));
free(dPtr);            // free(dPtr) has been added in the code to solve 2nd error
return array;
}
int main() {
List* array = calloc(1, sizeof(List));
array->data = calloc(10, sizeof(int32_t));
array = resizeArray(array);
                      // free(array->data); has been deleted to solve 1st error
free(array);
return 0;
}
```

After this I saved the file named **hw42b.c** and ran the following commands on terminal :-
1] **gcc -g hw42b.c**
2]**valgrind –leak-check=yes ./a.out**

The final output is given below :-

```
ritik@ritik-HP-Pavilion-Laptop-15-cs3xxx:~$ gcc -g hw42b.c
ritik@ritik-HP-Pavilion-Laptop-15-cs3xxx:~$ valgrind --leak-check=yes ./a.out
==19667== Memcheck, a memory error detector
==19667== Copyright (C) 2002-2017, and GNU GPL'd, by Julian Seward et al.
==19667== Using Valgrind-3.15.0 and LibVEX; rerun with -h for copyright info
==19667== Command: ./a.out
==19667==
==19667==
==19667== HEAP SUMMARY:
==19667==     in use at exit: 0 bytes in 0 blocks
==19667==   total heap usage: 3 allocs, 3 frees, 116 bytes allocated
==19667==
==19667== All heap blocks were freed -- no leaks are possible
==19667==
==19667== For lists of detected and suppressed errors, rerun with: -s
==19667== ERROR SUMMARY: 0 errors from 0 contexts (suppressed: 0 from 0)
ritik@ritik-HP-Pavilion-Laptop-15-cs3xxx:~$
```

As you can see in the output ERROR SUMMARY is :-
0 error from 0 contexts.

2c]

The code given in the question was saved in a file named **hw42c.c**.
After that the following commands were run in the terminal :-
1] **gcc -g hw42c.c**
2]**valgrind –leak-check=yes ./a.out**

The output is given below :-

```
ritik@ritik-HP-Pavilion-Laptop-15-cs3xxx:~$ gcc -g hw42c.c
ritik@ritik-HP-Pavilion-Laptop-15-cs3xxx:~$ valgrind --leak-check=yes ./a.out
==20300== Memcheck, a memory error detector
==20300== Copyright (C) 2002-2017, and GNU GPL'd, by Julian Seward et al.
==20300== Using Valgrind-3.15.0 and LibVEX; rerun with -h for copyright info
==20300== Command: ./a.out
==20300==
==20300== Invalid read of size 1
==20300==    at 0x1091B9: main (hw42c.c:7)
==20300==  Address 0x4a520d2 is 0 bytes after a block of size 34 alloc'd
==20300==    at 0x483B7F3: malloc (in /usr/lib/x86_64-linux-gnu/valgrind/vgpreload_memcheck-amd64-linux.so)
==20300==    by 0x109191: main (hw42c.c:5)
==20300==
==20300==
==20300== HEAP SUMMARY:
==20300==     in use at exit: 69 bytes in 2 blocks
==20300==   total heap usage: 2 allocs, 0 frees, 69 bytes allocated
==20300==
==20300== 34 bytes in 1 blocks are definitely lost in loss record 1 of 2
==20300==    at 0x483B7F3: malloc (in /usr/lib/x86_64-linux-gnu/valgrind/vgpreload_memcheck-amd64-linux.so)
==20300==    by 0x109191: main (hw42c.c:5)
==20300==
==20300== 35 bytes in 1 blocks are definitely lost in loss record 2 of 2
==20300==    at 0x483DD99: calloc (in /usr/lib/x86_64-linux-gnu/valgrind/vgpreload_memcheck-amd64-linux.so)
==20300==    by 0x109183: main (hw42c.c:4)
==20300==
==20300== LEAK SUMMARY:
==20300==    definitely lost: 69 bytes in 2 blocks
==20300==    indirectly lost: 0 bytes in 0 blocks
==20300==      possibly lost: 0 bytes in 0 blocks
==20300==    still reachable: 0 bytes in 0 blocks
==20300==         suppressed: 0 bytes in 0 blocks
==20300==
==20300== For lists of detected and suppressed errors, rerun with: -s
==20300== ERROR SUMMARY: 3 errors from 3 contexts (suppressed: 0 from 0)
ritik@ritik-HP-Pavilion-Laptop-15-cs3xxx:~$
```

There are **3 errors** in the given code :-
1] **Invalid read** :- The memory location that the process was trying to read is outside of the memory addresses that are available to the process.
2] **Memory Leak** of **definitely lost** type.
3] **Memory Leak** of **definitely lost** type.

Now to suppress the first error the following command should be used :-
1] **valgrind –leak-check=yes –gen-suppressions=yes ./a.out**
After running this command valgrind will stop after every error and ask us if we want to suppress the given error or not.
This is what we get at our terminals :-
—- Print suppression ? — [Return/N/n/Y/y/C/c] —-
To suppress 1st error we need to give **y**.
After that we again get the same message of print suppression. But this time we need to give **n** because we just want to suppress the 1st error.

We also get a code which we need to copy in a text file and save it.
The code is :-

```
==21194== ---- Print suppression ? --- [Return/N/n/Y/y/C/c] ---- y
{
   <insert_a_suppression_name_here>
   Memcheck:Addr1
   fun:main
}
```

Now I copied the above code in a text file and changed the 1st line of the code that is <**insert_a_suppression_name_here**> to **sup1**.
After that I saved the above text file as **s.supp**
Then the following command was used for suppression of 1st error :-
valgrind –leak-check=yes –suppressions=s.supp ./a.out
And the outcome of this command is given below :-

```
ritik@ritik-HP-Pavilion-Laptop-15-cs3xxx:~$ valgrind --leak-check=yes --suppressions=s.supp ./a.out
==23902== Memcheck, a memory error detector
==23902== Copyright (C) 2002-2017, and GNU GPL'd, by Julian Seward et al.
==23902== Using Valgrind-3.15.0 and LibVEX; rerun with -h for copyright info
==23902== Command: ./a.out
==23902==
==23902==
==23902== HEAP SUMMARY:
==23902==     in use at exit: 69 bytes in 2 blocks
==23902==   total heap usage: 2 allocs, 0 frees, 69 bytes allocated
==23902==
==23902== 34 bytes in 1 blocks are definitely lost in loss record 1 of 2
==23902==    at 0x483B7F3: malloc (in /usr/lib/x86_64-linux-gnu/valgrind/vgpreload_memcheck-amd64-linux.so)
==23902==    by 0x109191: main (hw42c.c:5)
==23902==
==23902== 35 bytes in 1 blocks are definitely lost in loss record 2 of 2
==23902==    at 0x483DD99: calloc (in /usr/lib/x86_64-linux-gnu/valgrind/vgpreload_memcheck-amd64-linux.so)
==23902==    by 0x109183: main (hw42c.c:4)
==23902==
==23902== LEAK SUMMARY:
==23902==    definitely lost: 69 bytes in 2 blocks
==23902==    indirectly lost: 0 bytes in 0 blocks
==23902==      possibly lost: 0 bytes in 0 blocks
==23902==    still reachable: 0 bytes in 0 blocks
==23902==         suppressed: 0 bytes in 0 blocks
==23902==
==23902== For lists of detected and suppressed errors, rerun with: -s
==23902== ERROR SUMMARY: 2 errors from 2 contexts (suppressed: 1 from 1)
ritik@ritik-HP-Pavilion-Laptop-15-cs3xxx:~$
```

The last line of the output is :-
ERROR SUMMARY: 2 errors from 2 contexts (suppressed: 1 from 1)
So we have successfully suppressed 1st error.

Now let's see the 2nd and 3rd error :-
According to the output we have errors at **line no. 4 and 5.**
It's a memory leak and to solve that we need to deallocate the memory. And for this we need to use the **free()** function.
After using free() function we need to again save the **hw42c.c** file.
Run the following command :-
1] **gcc -g hw42c.c**
2]**valgrind –leak-check=yes ./a.out**
But as we saved the file again so the suppression went out. So I again did the suppression of 1st error.
The output is given below :-

```
==24592== Memcheck, a memory error detector
==24592== Copyright (C) 2002-2017, and GNU GPL'd, by Julian Seward et al.
==24592== Using Valgrind-3.15.0 and LibVEX; rerun with -h for copyright info
==24592== Command: ./a.out
==24592==
==24592==
==24592== HEAP SUMMARY:
==24592==     in use at exit: 0 bytes in 0 blocks
==24592==   total heap usage: 2 allocs, 2 frees, 69 bytes allocated
==24592==
==24592== All heap blocks were freed -- no leaks are possible
==24592==
==24592== For lists of detected and suppressed errors, rerun with: -s
==24592== ERROR SUMMARY: 0 errors from 0 contexts (suppressed: 1 from 1)
ritik@ritik-HP-Pavilion-Laptop-15-cs3xxx:~$
```

Last line of the output :-
ERROR SUMMARY: 0 errors from 0 contexts (suppressed: 1 from 1)
We have suppressed 1st error and rectified 2nd and 3rd error successfully.

We can solve the 1st error as well :-
It was Invalid read of size 1 which occurred because if we see at the given code then we see that :-
source is given space to allocate 34 **char** but in the **for loop** source is taken till **35th memory which does not exists**.
Therefore to rectify this error we need to change the for loop given in the code with :-
for(int i=0; i<34; i++).
Final code with modifications is given below :-

```c
#include <stdlib.h>
#include <stdint.h>
int main() {
char* dest = calloc(35, sizeof(char));
char* source = malloc(34 * sizeof(char));
for(int i = 0; i < 34; i++) {        // i<35 has been changed to i<34 to solve 1st error
*(dest + i) = *(source + i);
}
free(dest);                          //free() function is used to solve 2nd and 3rd error.
free(source);

return 0;
}
```