UNIX
ASSIGNMENT-04

NAME-G.KALYAN RAM

ROLL NO:422150

SEC-A

QUESTION:

Generate different C programs that induce a segmentation fault error, select these examples of your choice, and employ the GDB utility for debugging on Linux.

Note:
1. Include multiple breakpoints while debugging
2. Upload your submission in a format consistent with the example provided in the material.

Example-1

```c
//fact.c
#include<stdio.h>

int factorial(int n) {
    if (n == 0 || n == 1) {
        printf("Factorial of %d is %d\n", n, 1);
        return 1;
    } else {
        int result = n * factorial(n - 1);
        printf("Factorial of %d is %d\n", n, result);
        return result;
    }
}// main.c


void main() {
    int n;
    printf("Enter the value of num:");
    scanf("%d",&n);
    factorial(n);

}
```

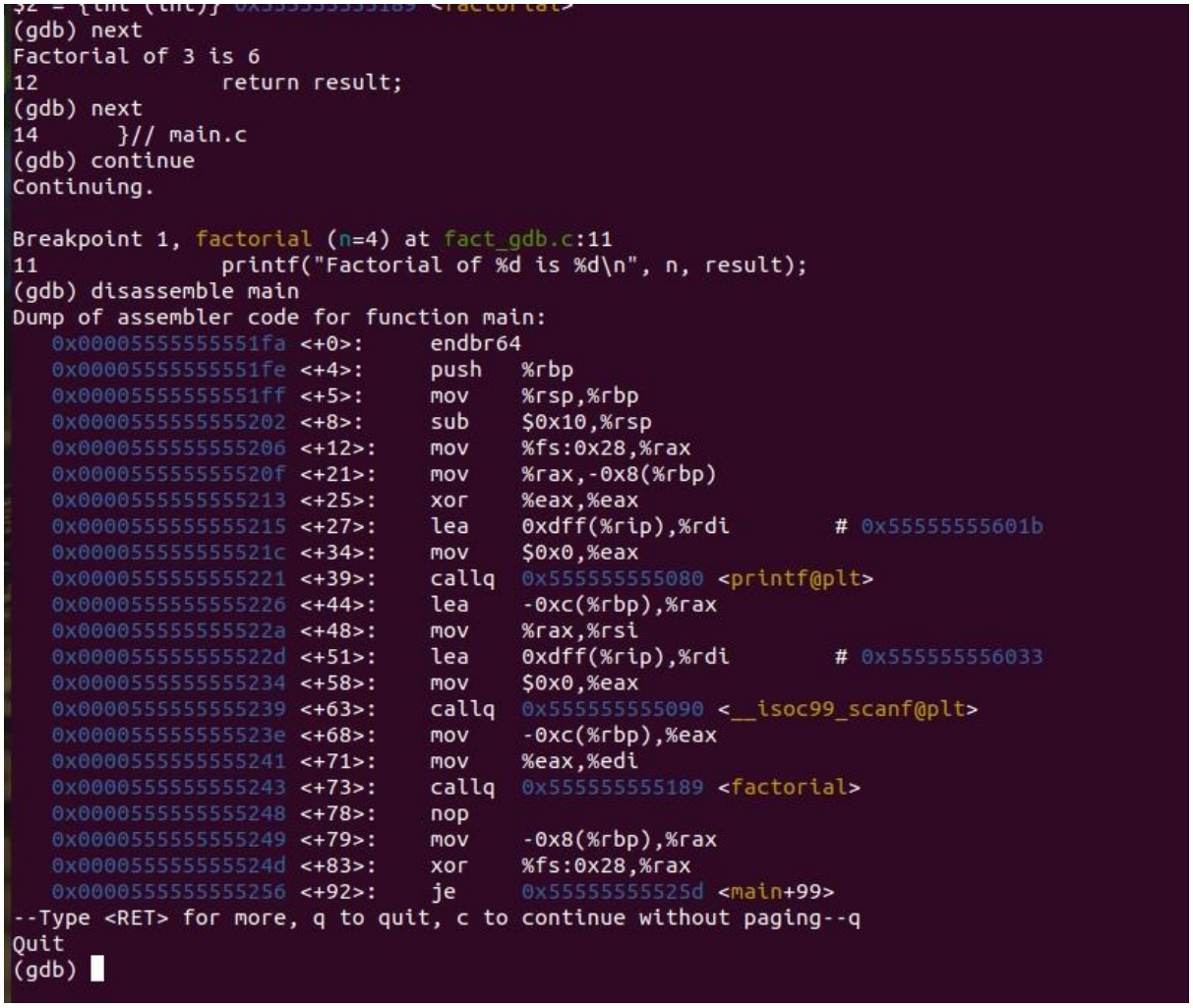
```
(base) student@welcome:~/Desktop/422150/unix-scripts/w6$ gcc -g fact_gdb.c
(base) student@welcome:~/Desktop/422150/unix-scripts/w6$ gdb ./a.out
GNU gdb (Ubuntu 9.2-0ubuntu1~20.04.1) 9.2
Copyright (C) 2020 Free Software Foundation, Inc.
License GPLv3+: GNU GPL version 3 or later <http://gnu.org/licenses/gpl.html>
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-linux-gnu".
Type "show configuration" for configuration details.
For bug reporting instructions, please see:
<http://www.gnu.org/software/gdb/bugs/>.
Find the GDB manual and other documentation resources online at:
    <http://www.gnu.org/software/gdb/documentation/>.

For help, type "help".
Type "apropos word" to search for commands related to "word"...
Reading symbols from ./a.out...
(gdb) run
Starting program: /home/student/Desktop/422150/unix-scripts/w6/a.out
Enter the value of num:6
Factorial of 1 is 1
Factorial of 2 is 2
Factorial of 3 is 6
Factorial of 4 is 24
Factorial of 5 is 120
Factorial of 6 is 720
[Inferior 1 (process 5003) exited normally]
(gdb) list
3       #include<stdio.h>
4
5       int factorial(int n) {
6           if (n == 0 || n == 1) {
7               printf("Factorial of %d is %d\n", n, 1);
8               return 1;
9           } else {
10              int result = n * factorial(n - 1);
11              printf("Factorial of %d is %d\n", n, result);
12              return result;
(gdb)
13          }
14      }// main.c
15
16
17      void main() {
18          int n;
```

```
13          }
14      }// main.c
15
16
17      void main() {
18          int n;
19          printf("Enter the value of num:");
20          scanf("%d",&n);
21          factorial(n);
22
(gdb)
23      }
24
(gdb) list
Line number 25 out of range; fact_gdb.c has 24 lines.
(gdb) break 11
Breakpoint 1 at 0x5555555551dc: file fact_gdb.c, line 11.
(gdb) run
Starting program: /home/student/Desktop/422150/unix-scripts/w6/a.out
Enter the value of num:6
Factorial of 1 is 1

Breakpoint 1, factorial (n=2) at fact_gdb.c:11
11              printf("Factorial of %d is %d\n", n, result);
(gdb) print i
No symbol "i" in current context.
(gdb) print n
$1 = 2
(gdb) next
Factorial of 2 is 2
12              return result;
(gdb) next
14      }// main.c
(gdb) next
factorial (n=3) at fact_gdb.c:10
10              int result = n * factorial(n - 1);
(gdb) next

Breakpoint 1, factorial (n=3) at fact_gdb.c:11
11              printf("Factorial of %d is %d\n", n, result);
(gdb) print factorial
$2 = {int (int)} 0x555555555189 <factorial>
(gdb) next
Factorial of 3 is 6
12              return result;
(gdb) next
```



Example-2

#insertion and deletion in a linked list

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

// Define the structure for a node in the linked list
typedef struct Node {
    int data;
    struct Node* next;
} Node;

// Function to create a new node
Node* createNode(int data) {
    Node* newNode = (Node*)malloc(sizeof(Node));
    if (newNode == NULL) {
        printf("Memory allocation failed\n");
        exit(1);
    }
    newNode->data = data;
```

```
    newNode->next = NULL;
    return newNode;
}

// Function to insert a new node at the beginning of the linked list
void insertAtBeginning(Node** head, int data) {
    Node* newNode = createNode(data);
    newNode->next = *head;
    *head = newNode;
}

// Function to display the linked list
void display(Node* head) {
    Node* temp = head;
    while (temp != NULL) {
        printf("%d -> ", temp->data);
        temp = temp->next;
    }
    printf("NULL\n");
}

// Function to delete a node with given key from the linked list
void deleteNode(Node** headRef, int key) {
    Node* temp = *headRef;
    Node* prev = NULL;

    // Introduce segmentation fault
    char* ptr = NULL;
    *ptr = 'x';

    // If head node itself holds the key to be deleted
    if (temp != NULL && temp->data == key) {
        *headRef = temp->next;
        free(temp);
        return;
    }

    // Search for the key to be deleted, keep track of the previous node as we
need to change 'prev->next'
    while (temp != NULL && temp->data != key) {
        prev = temp;
        temp = temp->next;
    }

    // If key was not present in linked list
    if (temp == NULL) {
        printf("Key not found in the linked list\n");
        return;
```

```
    }

    // Unlink the node from linked list
    prev->next = temp->next;
    free(temp);
}

int main() {
    Node* head = NULL;

    // Insert some elements into the linked list
    insertAtBeginning(&head, 5);
    insertAtBeginning(&head, 10);
    insertAtBeginning(&head, 15);

    // Display the linked list
    printf("Linked list: ");
    display(head);

    // Delete a node with key 10
    deleteNode(&head, 10);

    // Display the updated linked list
    printf("Linked list after deletion: ");
    display(head);

    return 0;
}
```

```
(base) student@welcome:~/Desktop/422150/unix-scripts$ gcc -g insertlink.c
(base) student@welcome:~/Desktop/422150/unix-scripts$ gdb ./a.out
GNU gdb (Ubuntu 9.2-0ubuntu1~20.04.1) 9.2
Copyright (C) 2020 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-linux-gnu".
Type "show configuration" for configuration details.
For bug reporting instructions, please see:
<http://www.gnu.org/software/gdb/bugs/>.
Find the GDB manual and other documentation resources online at:
    <http://www.gnu.org/software/gdb/documentation/>.

For help, type "help".
Type "apropos word" to search for commands related to "word"...
Reading symbols from ./a.out...
(gdb) run
Starting program: /home/student/Desktop/422150/unix-scripts/a.out
Linked list: 15 -> 10 -> 5 -> NULL

Program received signal SIGSEGV, Segmentation fault.
0x0000555555555303 in deleteNode (headRef=0x7fffffffdd30, key=10)
    at insertlink.c:46
46          *ptr = 'x';
(gdb) list
41          Node* temp = *headRef;
42          Node* prev = NULL;
43
44          // Introduce segmentation fault
45          char* ptr = NULL;
46          *ptr = 'x';
47
48          // If head node itself holds the key to be deleted
49          if (temp != NULL && temp->data == key) {
50              *headRef = temp->next;
(gdb)
51              free(temp);
52              return;
53          }
54
55          // Search for the key to be deleted, keep track of the previous node as we need to change 'prev->next'
56          while (temp != NULL && temp->data != key) {
57              prev = temp;
58              temp = temp->next;
```

```
58              temp = temp->next;
59          }
60
(gdb)
61          // If key was not present in linked list
62          if (temp == NULL) {
63              printf("Key not found in the linked list\n");
64              return;
65          }
66
67          // Unlink the node from linked list
68          prev->next = temp->next;
69          free(temp);
70      }
(gdb)
71
72      int main() {
73          Node* head = NULL;
74
75          // Insert some elements into the linked list
76          insertAtBeginning(&head, 5);
77          insertAtBeginning(&head, 10);
78          insertAtBeginning(&head, 15);
79
80          // Display the linked list
(gdb)
81          printf("Linked list: ");
82          display(head);
83
84          // Delete a node with key 10
85          deleteNode(&head, 10);
86
87          // Display the updated linked list
88          printf("Linked list after deletion: ");
89          display(head);
90
(gdb)
91          return 0;
92      }
93
(gdb) list
Line number 94 out of range; insertlink.c has 93 lines.
(gdb) break 45
Breakpoint 1 at 0x5555555552f7: file insertlink.c, line 45.
(gdb) break 46
Breakpoint 2 at 0x5555555552ff: file insertlink.c, line 46.
```

```
(gdb) list
Line number 94 out of range; insertlink.c has 93 lines.
(gdb) break 45
Breakpoint 1 at 0x5555555552f7: file insertlink.c, line 45.
(gdb) break 46
Breakpoint 2 at 0x5555555552ff: file insertlink.c, line 46.
(gdb) print head
No symbol "head" in current context.
(gdb) print temp
$1 = (Node *) 0x5555555592e0
(gdb) next

Program terminated with signal SIGSEGV, Segmentation fault.
The program no longer exists.
(gdb) quit
(base) student@welcome:~/Desktop/422150/unix-scripts$
```