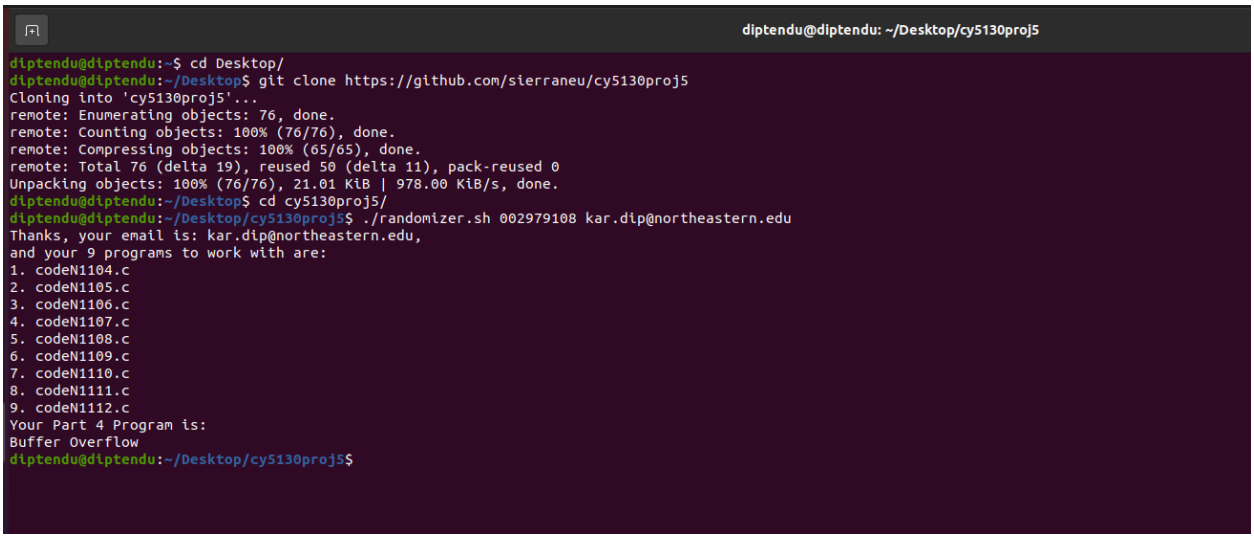
Part 0



Part 1

```
#!/bin/bash

# ARG COUNT CHECK - NUID AND EMAIL

if [[ $# -ne 2 ]]; then

   echo "Usage: $0 <nuid> <@northeastern email id>"

   exit 1

fi

# CHECK NUID PATTERN

if [[ ! $1 =~ ^00[0-9]{7}$ ]]; then

   echo "Invalid NUID. Expected 9 digits starting with two zeros e.g. 002979108."

   exit 1

fi

# CHECK NORTHEASTERN EMAIL PATTERN - assuming all email are lowercase otherwise  $2 =~ ^[a-zA-Z]+(\.[a-zA-Z]+[0-9]*)\@(northeastern|neu)\.edu$

if [[ ! $2 =~ ^[a-z]+(\.[a-z]+[0-9]*)\@(northeastern|neu)\.edu$ ]]; then

   echo "Invalid northeastern email."

   exit 1

fi
```

```
# GIVEN CODE HERE
nuid=$((10#$1))
sc1=$(( $nuid % 19))
echo -e "Thanks, your email is: $2, \nand your 9 programs to work with are:"
for (( c=1; c<10; c++ ))
do
	increment=$(( $c))
	code=$(($(($sc1 + $increment)) % 19))
	code=$(printf %02d $code)
	echo "$c. codeN11$code.c"
done

sc2=$(( $nuid % 3))
echo -e "Your Part 4 Program is:"
if [[ $sc2 -eq 0 ]]
then
echo "Buffer Overflow"
fi
if [[ $sc2 -eq 1 ]]
then
echo "Undefined Behavior"
fi
if [[ $sc2 -eq 2 ]]
then
echo "Memory Leak"
fi
```

NUID Pattern – Starts with 00, followed by anything from [0-9]. {7} means exactly 7 times.

Email Pattern – [a-z]+ means any lowercase 1 or more times. Next match the character ".", then again [a-z]+ means any lowercase 1 or more times. Then [0-9]* means any digits 0 or more times as digits can or cannot be present. Next match the character "@" followed by either the string "northeastern" or "neu" denoted by "|" and finally ".edu".

Part 2: Coding and Running the Analysis

1. codeN1104.c

```
diptendu@diptendu:~/Desktop/project5$ scan-build -o . gcc codeN1104.c
scan-build: Using '/usr/lib/llvm-10/bin/clang' for static analysis
codeN1104.c:30:5: warning: 3rd function call argument is an uninitialized value
    printf("Team with max revenue (%d) is: %s\n", maxRevenue, maxRevenueTeam);
    ^~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
1 warning generated.
scan-build: 1 bug found.
scan-build: Run 'scan-view /home/diptendu/Desktop/project5/2023-04-07-103847-4498-1' to examine bug reports.
diptendu@diptendu:~/Desktop/project5$ ./a.out
Arsenal: -14528
Barcelona: -21120
Paris St: -6496
Team with max revenue (0) is: (null)
diptendu@diptendu:~/Desktop/project5$
```

project5 - scan-build results

User:	diptendu@diptendu
Working Directory:	/home/diptendu/Desktop/project5
Command Line:	gcc codeN1104.c
Clang Version:	clang version 10.0.0-4ubuntu1
Date:	Fri Apr 7 10:38:47 2023

Bug Summary

Bug Type	Quantity	Display?
All Bugs	1	☑
Logic error		
Uninitialized argument value	1	☑

Reports

Bug Group	Bug Type ▾	File	Function/Method	Line	Path Length	
Logic error	Uninitialized argument value	codeN1104.c	main	30	12	View Report

```
int main() {

    struct SoccerTeam teams[] = {arsenalFC, barcelonaFC, parisSG};
    short maxRevenue = 0;
    char *maxRevenueTeam;
```

1 'maxRevenueTeam' declared without an initial value →

```
    for (int i = 0; i < 3; i++) {
```

2 ← Loop condition is true. Entering loop body →

4 ← Loop condition is true. Entering loop body →

7 ← Loop condition is true. Entering loop body →

10 ← Loop condition is false. Execution continues on line 30 →

```
        long totalRevenue = calculateSoccerRevenue(teams[i].revenueOverall, teams[
        if (totalRevenue > maxRevenue) {
```

3 ← Taking false branch →

5 ← Assuming 'totalRevenue' is <= 'maxRevenue' →

6 ← Taking false branch →

8 ← Assuming 'totalRevenue' is <= 'maxRevenue' →

9 ← Taking false branch →

```
            maxRevenue = totalRevenue;
            maxRevenueTeam = teams[i].name;
        }
        printf("%s: %ld\n", teams[i].name, totalRevenue);
    }
    printf("Team with max revenue (%d) is: %s\n", maxRevenue, maxRevenueTeam);
```

11 ← 3rd function call argument is an uninitialized value

```
    return (0);
}
```

Problems identified with clang:

codeN1104.c:30:5: warning: 3rd function call argument is an uninitialized value.

printf("Team with max revenue (%d) is: %s\n", maxRevenue, maxRevenueTeam);

This is due to char *maxRevenueTeam; which is not initialized.

Problems identified from output:

short calculateSoccerRevenue(long revenueTillDate, short int revenueThisYear) {

return revenueTillDate + revenueThisYear;

}

The return type of calculateSoccerRevenue is short, the arguments will add up to numbers that are more than the upper limit of short, hence there will be an overflow and so we see the negative values.

Same for the argument - short int revenueThisYear

And same for short maxRevenue = 0;

```
 long totalRevenue = calculateSoccerRevenue(teams[i].revenueOverall, teams[i].revenueThisYear);
if (totalRevenue > maxRevenue) {
        maxRevenue = totalRevenue;
        maxRevenueTeam = teams[i].name;
    }
```

So, we are assigning long to short, causing overflow / wrap-around.

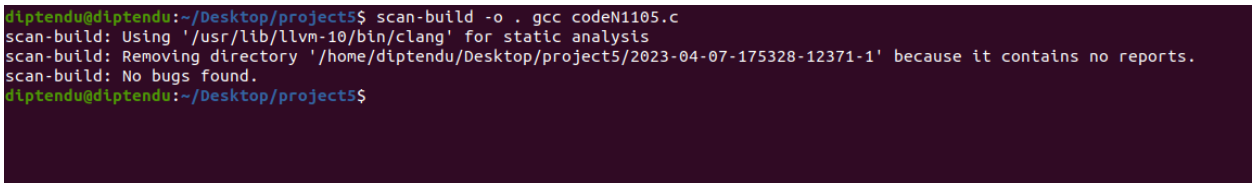
Fixed Code:

```
#include <stdio.h>

struct SoccerTeam {
   char name[50];
   int revenueThisYear;
   long revenueOverall;
};

struct SoccerTeam arsenalFC = {.name = "Arsenal", .revenueOverall = 12300000, .revenueThisYear = 2300000};
struct SoccerTeam barcelonaFC = {.name = "Barcelona", .revenueOverall = 1220000, .revenueThisYear = 9900000};
struct SoccerTeam parisSG = {.name = "Paris St", .revenueOverall = 1220000, .revenueThisYear = 55200000};

// changed ret type and revenueThisYear due to overflow from long to short assignment
long calculateSoccerRevenue(long revenueTillDate, int revenueThisYear) {
   return revenueTillDate + revenueThisYear;
}
```

```
int main() {
   struct SoccerTeam teams[] = {arsenalFC, barcelonaFC, parisSG};
   // variable type from short to long
   long maxRevenue = 0;
     // initialized maxRevenueTeam
   char *maxRevenueTeam = NULL;
   for (int i = 0; i < 3; i++) {
      long totalRevenue = calculateSoccerRevenue(teams[i].revenueOverall,
teams[i].revenueThisYear);
      if (totalRevenue > maxRevenue) {
         maxRevenue = totalRevenue;
         maxRevenueTeam = teams[i].name;
      }
      printf("%s: %ld\n", teams[i].name, totalRevenue);
   }
   // %d to %ld to correctly assign long type
   printf("Team with max revenue (%ld) is: %s\n", maxRevenue, maxRevenueTeam);
   return (0);
}
```

Scan-build after fix

2. codeN1105.c



Clang does not show any issues.
There are some issues with the password XOR operation.
Encryption and decryption modifying the original data. Instead of modifying the input, create new character arrays to store the encrypted or decrypted values. But I think it is fine for a single run but running it on a server may be a problem.
Also, there is a case where the password provided when XORED with the secret will be NULL. If one of the strings passed to strcmp() is NULL, then the behavior is undefined, which means that the result is unpredictable and can vary depending on the compiler and platform being used.
Same is valid for username if username is NULL check is not there.
Not going into the fact that stored passwords can be recovered from the binary.

Fixed code

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

#define MAX_PASSWORD_LEN 9

struct User {
   char username[50];
   char password[MAX_PASSWORD_LEN];
   int uid;
};

struct User usersDb[3] = {
      {.username = "admin", .password = "jrAz+"},
      {.username = "userA", .password = "ocAr6"},
      {.username = "userB", .password = "idMf("},
};

const char secret[8] = {11, 22, 44, 19, 69, 30, 90, 14};

// we can change the method singature to include a copy array like decrypt, since it is not used not changing
char *encrypt(char *toEncrypt, int size) {
   int i;
   for (i = 0; i < size; i++)
```

```
        toEncrypt[i] ^= secret[i];

    return toEncrypt;
}


// changed method singature to include a copy array where comparison will be done
void decrypt(char *toDecrypt, char *decrypted, int size) {
    for (int i = 0; i < size; i++)
        decrypted[i] = toDecrypt[i] ^ secret[i];

}

int checkValidUserName(char *username) {

       // not null check
       if (username != NULL) {
    for (int i = 0; i < 3; i++) {
        if (strcmp(username, usersDb[i].username) == 0) {
            return 1;
        }
    }
}
    return 0;
}

int checkValidPassword(char *username, char *password) {
    char decryptedPassword[MAX_PASSWORD_LEN];
    for (int i = 0; i < 3; i++) {
        if (strcmp(username, usersDb[i].username) == 0) {
            decrypt(usersDb[i].password, decryptedPassword, strlen(usersDb[i].password));
            // not null check
            if (password != NULL || decryptedPassword != NULL) {
            if (strcmp(password, decryptedPassword) == 0) {
                return 1;
            }
                    }
        }
    }
    return 0;
}

int checkCredentials(char *username, char *password) {
    if (checkValidUserName(username)) {
```

```
        if (checkValidPassword(username, password)) {
            printf("Welcome %s", username);
            return 1;
        } else {
            printf("%s", "Password is invalid");
            return 0;
        }
    } else {
        printf("%s", "Username is invalid");
    }
    return 1;
}

int main(int argc, char **argv) {
    char username[20];
    char password[20];
    printf("Enter username:");
    scanf("%19s", username);
    printf("Enter password");
    scanf("%19s", password);
    checkCredentials(username, password);
}
```

3. codeN1106.c

```
diptendu@diptendu:~/Desktop/project5$ scan-build -o . gcc codeN1106.c
scan-build: Using '/usr/lib/llvm-10/bin/clang' for static analysis
codeN1106.c:13:25: warning: Dereference of null pointer (loaded from variable 'rtp1')
    for (int i = 0; i < (*rtp1); i++) {
                        ^~~~~~~
1 warning generated.
scan-build: 1 bug found.
scan-build: Run 'scan-view /home/diptendu/Desktop/project5/2023-04-07-123231-7271-1' to examine bug reports.
diptendu@diptendu:~/Desktop/project5$ ./a.out
Segmentation fault (core dumped)
diptendu@diptendu:~/Desktop/project5$
```

```c
// Use C-Lang Static Analyzer to analyze and fix the bugs
// Also review the output generated to find any bugs that the static analyzer misses.
// Hint this code has 3 bugs that can be identified using static analyzer, 1 vulnerability wi

//MACROS
#include<stdio.h>

# define ARRAY_MEMORY_SIZE 10

// custom method to print digits from 11 to value of size in main().
void method(int *rtp, int *rtp1) {
    int count;
    for (int i = 0; i < (*rtp1); i++) {
                    4 ← Dereference of null pointer (loaded from variable 'rtp1')
        printf("Numbers are: %d \n", (*rtp + count));
        count++;
    }

}

// custom method to fill an array with numbers till valid defined "macro' size
void memory_filler(int arr[]) {
    int inc = 0;
    for (int k = 0; k < 30; k++) {
        arr[k] = inc;
        inc += 1;
        printf("Array element is: %d \n", arr[k]);
    }

}


int main() {
    // variable initializations

    int first_number = 10;
    int size = 9;
    int random_array[ARRAY_MEMORY_SIZE];

    //Note for developers: use as size in method using defined pointers (not directly)

    int *ptr_to_first_number = NULL;
    int *ptr_to_size = NULL;
    1 'ptr_to_size' initialized to a null pointer value →

    // Hint:: add the assign the references to the pointers here:


    //function call statements
    method(ptr_to_first_number, ptr_to_size);
                        2 ← Passing null pointer value via 2nd parameter 'rtp1' →

    3 ← Calling 'method' →
    memory_filler(random_array);

    return 0;

}
```

1st fix - int *ptr_to_size = &size;

Now clang output

```
diptendu@diptendu:~/Desktop/project5$ scan-build -o . gcc codeN1106.c
scan-build: Using '/usr/lib/llvm-10/bin/clang' for static analysis
codeN1106.c:14:39: warning: Dereference of null pointer (loaded from variable 'rtp')
        printf("Numbers are: %d \n", (*rtp + count));
                                      ^~~~
1 warning generated.
scan-build: 1 bug found.
scan-build: Run 'scan-view /home/diptendu/Desktop/project5/2023-04-07-124219-7721-1' to examine bug reports.
diptendu@diptendu:~/Desktop/project5$ ./a.out
Segmentation fault (core dumped)
diptendu@diptendu:~/Desktop/project5$
```

2nd fix - int *ptr_to_first_number = &first_number;

Now clang output

```
scan-build: Using '/usr/lib/llvm-10/bin/clang' for static analysis
codeN1106.c:14:44: warning: The right operand of '+' is a garbage value
        printf("Numbers are: %d \n", (*rtp + count));
                                           ^ ~~~~~
1 warning generated.
scan-build: 1 bug found.
scan-build: Run 'scan-view /home/diptendu/Desktop/project5/2023-04-07-124607-7816-1' to examine bug reports.
diptendu@diptendu:~/Desktop/project5$ ./a.out
Numbers are: 67108874
Numbers are: 67108875
Numbers are: 67108876
Numbers are: 67108877
Numbers are: 67108878
Numbers are: 67108879
Numbers are: 67108880
Numbers are: 67108881
Numbers are: 67108882
Array element is: 0
Array element is: 1
Array element is: 2
Array element is: 3
Array element is: 4
Array element is: 5
Array element is: 6
Array element is: 7
Array element is: 8
Array element is: 9
Array element is: 10
Array element is: 11
Array element is: 12
Array element is: 13
Array element is: 14
Array element is: 15
Array element is: 16
Array element is: 17
Array element is: 18
Array element is: 19
Array element is: 20
Array element is: 21
Array element is: 22
Array element is: 23
Array element is: 24
Array element is: 25
Array element is: 26
Array element is: 27
Array element is: 28
Array element is: 29
*** stack smashing detected ***: terminated
Aborted (core dumped)
diptendu@diptendu:~/Desktop/project5$
```

3rd fix - int count=0;

```
diptendu@diptendu:~/Desktop/project5$ scan-build -o . gcc codeN1106.c
scan-build: Using '/usr/lib/llvm-10/bin/clang' for static analysis
scan-build: Removing directory '/home/diptendu/Desktop/project5/2023-04-07-124812-7907-1' because it contains no reports.
scan-build: No bugs found.
diptendu@diptendu:~/Desktop/project5$ ./a.out
Numbers are: 10
Numbers are: 11
Numbers are: 12
Numbers are: 13
Numbers are: 14
Numbers are: 15
Numbers are: 16
Numbers are: 17
Numbers are: 18
Array element is: 0
Array element is: 1
Array element is: 2
Array element is: 3
Array element is: 4
Array element is: 5
Array element is: 6
Array element is: 7
Array element is: 8
Array element is: 9
Array element is: 10
Array element is: 11
Array element is: 12
Array element is: 13
Array element is: 14
Array element is: 15
Array element is: 16
Array element is: 17
Array element is: 18
Array element is: 19
Array element is: 20
Array element is: 21
Array element is: 22
Array element is: 23
Array element is: 24
Array element is: 25
Array element is: 26
Array element is: 27
Array element is: 28
Array element is: 29
*** stack smashing detected ***: terminated
Aborted (core dumped)
diptendu@diptendu:~/Desktop/project5$
```

Problems identified with clang:

2 null pointer derefence

int *ptr_to_first_number = &first_number;

int *ptr_to_size = &size;

were set to NULL

in void method – count was not initialized.

Problems identified from output:

memory_filler(random_array);

int random_array[ARRAY_MEMORY_SIZE];

ARRAY_MEMORY_SIZE is 10 but in the method, the loop is till 30.

So it is better to fix the loop to ARRAY_MEMORY_SIZE instead of 30,

Final fixed code:

```
//MACROS
#include<stdio.h>
# define ARRAY_MEMORY_SIZE 10

// custom method to print digits from 11 to value of size in main().
void method(int *rtp, int *rtp1) {
        // fixed initialization of count
   int count=0;
   for (int i = 0; i < (*rtp1); i++) {
      printf("Numbers are: %d \n", (*rtp + count));
      count++;
   }

}

// custom method to fill an array with numbers till valid defined "macro' size
void memory_filler(int arr[]) {
   int inc = 0;
   // was looping to 30 but arr size defined by macro is 10
   for (int k = 0; k < ARRAY_MEMORY_SIZE; k++) {
      arr[k] = inc;
      inc += 1;
      printf("Array element is: %d \n", arr[k]);
   }

}
```

```
int main() {
   // variable initializations
         // need to print from 11 OR we can put 10 here are set count to 1, either works
   int first_number = 11;
   int size = 9;
   int random_array[ARRAY_MEMORY_SIZE];

   //Note for developers: use as size in method using defined pointers (not directly)
         // set the initial value of pointers to avoid null dereference
   int *ptr_to_first_number = &first_number;
   int *ptr_to_size = &size;
   // Hint:: add the assign the references to the pointers here:
   //function call statements
   method(ptr_to_first_number, ptr_to_size);
   memory_filler(random_array);

   return 0;

}
```

Scan-build after fix

4. codeN1107.c

```
diptendu@diptendu:~/Desktop/project5$ scan-build -o . gcc codeN1107.c
scan-build: Using '/usr/lib/llvm-10/bin/clang' for static analysis
codeN1107.c:12:49: warning: Dereference of null pointer (loaded from variable 'ptr1')
    printf("Original First Number is : %d \n ", (*ptr1));
                                                ^~~~~~~
1 warning generated.
scan-build: 1 bug found.
scan-build: Run 'scan-view /home/diptendu/Desktop/project5/2023-04-07-144002-8528-1' to examine bug reports.
diptendu@diptendu:~/Desktop/project5$ ./a.out
Segmentation fault (core dumped)
diptendu@diptendu:~/Desktop/project5$
```

```c
// Use C-Lang Static Analyzer to analyze and fix the bugs
// Also review the output generated to find any bugs that the static analyzer misses
// Hint this code has 1-2 bugs that can be identified using static analyzer, 2 Tasks
//1.. add null pointer check condition in the swap method for swapping logic.
//2.. Complete swapping Logic using given pointers.

//MACROS
#include<stdio.h>

// custom method to swap digits (without using third variable/pointer) passed to it
void swap(int *ptr1, int *ptr2) {
    printf("Original First Number is : %d \n ", (*ptr1));
                                    4 ← Dereference of null pointer (loaded from variable 'ptr1')
    printf("Swapped Second Number is : %d \n ", (*ptr2));

    // ---------------------- Secure Coding Task -----------------------------///
    // Task 1: Implement a NULL pointer check condition (use if statment) (most impo

    // ------  add null pointer check condition Here-------

    // logic for swapping
    *ptr1 = *ptr1 + *ptr2;
    //Task 2 : complete logic for swapping (Hint: required statements 2)
    //----- add logic here ------

    //print statements
    printf("Swapped First Number is : %d \n", (*ptr1));
    printf("Swapped Second Number is : %d \n", (*ptr2));

}

int main() {
    // variable initializations
    int first_number = 10;
    int second_number = 20;

    //Note for developers(students): use numbers in method using defined pointers (n

    int *ptr_to_first_number = NULL;
    1 'ptr_to_first_number' initialized to a null pointer value →
    int *ptr_to_second_number = NULL;
    // Hint:: add the assign the references to the pointers here:


    //function call statements
    swap(ptr_to_first_number, ptr_to_second_number);
        2 ← Passing null pointer value via 1st parameter 'ptr1' →
    3 ← Calling 'swap' →

    return 0;

}
```

1st fix

int *ptr_to_first_number = &first_number;

Now clang output

```
diptendu@diptendu:~/Desktop/project5$ scan-build -o . gcc codeN1107.c
scan-build: Using '/usr/lib/llvm-10/bin/clang' for static analysis
codeN1107.c:13:49: warning: Dereference of null pointer (loaded from variable 'ptr2')
    printf("Swapped Second Number is : %d \n ", (*ptr2));
                                                ^~~~~~
1 warning generated.
scan-build: 1 bug found.
scan-build: Run 'scan-view /home/diptendu/Desktop/project5/2023-04-07-144449-8682-1' to examine bug reports.
diptendu@diptendu:~/Desktop/project5$ ./a.out
Original First Number is : 10
Segmentation fault (core dumped)
diptendu@diptendu:~/Desktop/project5$
```

2nd fix

int *ptr_to_second_number = &second_number;

now clang output

```
diptendu@diptendu:~/Desktop/project5$ scan-build -o . gcc codeN1107.c
scan-build: Using '/usr/lib/llvm-10/bin/clang' for static analysis
scan-build: Removing directory '/home/diptendu/Desktop/project5/2023-04-07-144601-8774-1' because it contains no reports.
scan-build: No bugs found.
diptendu@diptendu:~/Desktop/project5$ ./a.out
Original First Number is : 10
 Swapped Second Number is : 20
 Swapped First Number is : 30
Swapped Second Number is : 20
diptendu@diptendu:~/Desktop/project5$
```

Problems identified with clang:

2 null pointer derefence

int *ptr_to_first_number = &first_number;

int *ptr_to_second_number = &second_number;

these were NULL

Problems identified with output

Swapping not implemented.

Final fixed code

```
// Use C-Lang Static Analyzer to analyze and fix the bugs

// Also review the output generated to find any bugs that the static analyzer misses.

// Hint this code has 1-2 bugs that can be identified using static analyzer, 2 Tasks will have to be completed

//1.. add null pointer check condition in the swap method for swapping logic.

//2.. Complete swapping Logic using given pointers.


//MACROS

#include<stdio.h>


// custom method to swap digits (without using third variable/pointer) passed to it as arguments from main()

void swap(int *ptr1, int *ptr2) {


        // null pointer check here

        if (ptr1 == NULL || ptr2 == NULL) {

    printf("Error: One or both pointers are NULL.\n");

    return;

  }

  printf("Original First Number is : %d \n ", (*ptr1));

  // print statement swapped -> original

  printf("Original Second Number is : %d \n ", (*ptr2));


  // ---------------------- Secure Coding Task ------------------------////

  // Task 1: Implement a NULL pointer check condition (use if statment) (most important secure coding practice)


  // ------  add null pointer check condition Here-------
```

```
    // logic for swapping
    *ptr1 = *ptr1 + *ptr2;
    // adding rest of swap code
    *ptr2 = *ptr1 - *ptr2;
    *ptr1 = *ptr1 - *ptr2;

    //Task 2 : complete logic for swapping (Hint: required statements 2)
    //----- add logic here ------

    //print statements
    printf("Swapped First Number is : %d \n", (*ptr1));
    printf("Swapped Second Number is : %d \n", (*ptr2));

}

int main() {
    // variable initializations
    int first_number = 10;
    int second_number = 20;

    //Note for developers(students): use numbers in method using defined pointers (not directly)

            // initialized the pointers
    int *ptr_to_first_number = &first_number;
    int *ptr_to_second_number = &second_number;
    // Hint:: add the assign the references to the pointers here:


    //function call statements
```

```
    swap(ptr_to_first_number, ptr_to_second_number);


    return 0;


}
```

Scan build after fix

```
diptendu@diptendu:~/Desktop/project5$ scan-build -o . gcc codeN1107.c
scan-build: Using '/usr/lib/llvm-10/bin/clang' for static analysis
scan-build: Removing directory '/home/diptendu/Desktop/project5/2023-04-07-145059-8935-1' because it contains no reports.
scan-build: No bugs found.
diptendu@diptendu:~/Desktop/project5$ ./a.out
Original First Number is : 10
 Original Second Number is : 20
 Swapped First Number is : 20
Swapped Second Number is : 10
diptendu@diptendu:~/Desktop/project5$
```

5. codeN1108.c

```
diptendu@diptendu:~/Desktop/project5$ scan-build -o . gcc codeN1108.c
scan-build: Using '/usr/lib/llvm-10/bin/clang' for static analysis
codeN1108.c: In function ‘even_number’:
codeN1108.c:24:9: error: expected expression before ‘)’ token
   24 |     if ()//add logic for checking even number inside if condition ()
      |        ^
codeN1108.c: In function ‘main’:
codeN1108.c:57:26: warning: implicit declaration of function ‘malloc’ [-Wimplicit-function-declaration]
   57 |     ptr_to_arr = (int *) malloc(sizeof(double));
      |                          ^~~~~~
codeN1108.c:57:26: warning: incompatible implicit declaration of built-in function ‘malloc’
codeN1108.c:10:1: note: include ‘<stdlib.h>’ or provide a declaration of ‘malloc’
    9 | #include<stdio.h>
  +++ |+#include <stdlib.h>
   10 |
scan-build: Removing directory '/home/diptendu/Desktop/project5/2023-04-07-152644-9118-1' because it contains no reports.
scan-build: No bugs found.
diptendu@diptendu:~/Desktop/project5$
```

Included the stdlib and added the even number check - if (*ptr1 % 2 == 0)

Clang output now

```
diptendu@diptendu:~/Desktop/project5$ scan-build -o . gcc codeN1108.c
scan-build: Using '/usr/lib/llvm-10/bin/clang' for static analysis
codeN1108.c:15:49: warning: Dereference of null pointer (loaded from variable 'ptr1')
    printf("Original First Number is : %d \n ", (*ptr1));
                                                 ^~~~~~
codeN1108.c:59:26: warning: Result of 'malloc' is converted to a pointer of type 'int', which is incompatible with sizeof operand type 'double'
    ptr_to_arr = (int *) malloc(sizeof(double));
                 ~~~~~  ^~~~~~ ~~~~~~~~~~~~~~
2 warnings generated.
scan-build: 2 bugs found.
scan-build: Run 'scan-view /home/diptendu/Desktop/project5/2023-04-07-152907-9268-1' to examine bug reports.
diptendu@diptendu:~/Desktop/project5$
```

Fixes

int *ptr_to_first_number = &first_number;

ptr_to_arr = (int *) malloc(sizeof(int));

Clang output now

```
diptendu@diptendu:~/Desktop/project5$ scan-build -o . gcc codeN1108.c
scan-build: Using '/usr/lib/llvm-10/bin/clang' for static analysis
codeN1108.c:16:34: warning: Dereference of null pointer (loaded from variable 'ptr2')
    printf("Choice is : %c \n ", (*ptr2));
                                  ^~~~~~
1 warning generated.
scan-build: 1 bug found.
scan-build: Run 'scan-view /home/diptendu/Desktop/project5/2023-04-07-153345-9498-1' to examine bug reports.
diptendu@diptendu:~/Desktop/project5$
```

Fixes

Clang output now

```
diptendu@diptendu:~/Desktop/project5$ scan-build -o . gcc codeN1108.c
scan-build: Using '/usr/lib/llvm-10/bin/clang' for static analysis
codeN1108.c:15:5: warning: 2nd function call argument is an uninitialized value
    printf("Original First Number is : %d \n ", (*ptr1));
    ^~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
1 warning generated.
scan-build: 1 bug found.
scan-build: Run 'scan-view /home/diptendu/Desktop/project5/2023-04-07-153610-9635-1' to examine bug reports.
diptendu@diptendu:~/Desktop/project5$
```

Fixes

int first_number=10;

Clang output now

```
diptendu@diptendu:~/Desktop/project5$ scan-build -o . gcc codeN1108.c
scan-build: Using '/usr/lib/llvm-10/bin/clang' for static analysis
codeN1108.c:68:5: warning: 2nd function call argument is an uninitialized value
    array_print(ptr_to_arr, size_array);
    ^~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
1 warning generated.
scan-build: 1 bug found.
scan-build: Run 'scan-view /home/diptendu/Desktop/project5/2023-04-07-153721-9724-1' to examine bug reports.
diptendu@diptendu:~/Desktop/project5$
```

Fixes

int size_array=10;

Clang output now

```
diptendu@diptendu:~/Desktop/project5$ scan-build -o . gcc codeN1108.c
scan-build: Using '/usr/lib/llvm-10/bin/clang' for static analysis
scan-build: Removing directory '/home/diptendu/Desktop/project5/2023-04-07-153800-9794-1' because it contains no reports.
scan-build: No bugs found.
diptendu@diptendu:~/Desktop/project5$
```

Apart from this

Implemented the null checks

Added a free before return

Fixed Code

```
// Use C-Lang Static Analyzer to analyze and fix the bugs

// Also review the output generated to find any bugs that the static analyzer misses.

// Hint this code has 5-6 bugs that can be identified using static analyzer, 3 Tasks will also have to be completed

//1.. add null pointer check condition for ptr1 and ptr2 in the even_number method

//2.. add null pointer check condition for ptr3 in the array_print method

//3.. Complete even Logic using given pointer.


//MACROS (Hint: 1 macro definition for a library function is missing (look at warnings maybe?))

#include<stdio.h>

// stdlib for malloc

#include<stdlib.h>



// custom method to check for even or odd number

void even_number(int *ptr1, char *ptr2) {


        // added null pointer check

        if (ptr1 == NULL || ptr2 == NULL) {

    printf("Error: One or both pointers are NULL.\n");

    return;

  }


  printf("Original First Number is : %d \n ", (*ptr1));

  printf("Choice is : %c \n ", (*ptr2));


  // ---------------------- Secure Coding Task ------------------------------////
```

```
   // Task 1: Implement a NULL pointer check condition (use if statment) (most important secure coding practice)


   // ------  add null pointer check condition for ptr1 Here-------


   //Task 2 : complete logic for detecting an even number


           // added even num check
   if (*ptr1 % 2 == 0)//add logic for checking even number inside if condition ()
   {
      //print statements
      // printf typo - Numbner ->Number
      printf("Number %d is Even : \n", (*ptr1));
      printf("Confirmed choice is : %c \n", (*ptr2));
   } else {
      printf("Odd Number\n");
   }


}


//another custom array printing an array
void array_print(int *ptr3, int n) {
           // added null pointer check
           if (ptr3 == NULL) {
      printf("Error: Pointer is NULL.\n");
      return;
   }
```

```
    // Task 3:: Add code here (if check condition for NULL pointer)
    for (int i = 0; i < n + 1; i++) {
        ptr3[i] = i;
        printf("Array is:%d\n ", ptr3[i]);
    }


}

int main() {
    // variable initializations (Hint: Missing values?????)
    // set the first_number to check the provided number even/odd
    int first_number=11;
    char choice = 'Y';
    int *ptr_to_arr;
    // initialized size array to a value
    int size_array=10;

    //Note for developers(students): use variables in method using defined pointers (not directly)
            // initialized the pointers to avoid null derefence check
    int *ptr_to_first_number = &first_number;
    char *ptr_to_char = &choice;
    // incompatible operand type double to int
    ptr_to_arr = (int *) malloc(sizeof(int));
    // Hint:: add and assign the references to the pointers here:


    //function call statements
    even_number(ptr_to_first_number, ptr_to_char);
```

```
    array_print(ptr_to_arr, size_array);


    // free the malloc call
    free(ptr_to_arr);


    return 0;


}
```

Scan build after fix

6. codeN1109.c

```
diptendu@diptendu:~/Desktop/project5$ scan-build -o . gcc codeN1109.c
scan-build: Using '/usr/lib/llvm-10/bin/clang' for static analysis
codeN1109.c: In function ‘main’:
codeN1109.c:32:24: warning: implicit declaration of function ‘malloc’ [-Wimplicit-function-declaration]
   32 |     ptr_to_arr = (int*)malloc(sizeof(char));
      |                        ^~~~~~
codeN1109.c:32:24: warning: incompatible implicit declaration of built-in function ‘malloc’
codeN1109.c:7:1: note: include ‘<stdlib.h>’ or provide a declaration of ‘malloc’
    6 | #include<stdio.h>
  +++ |+#include <stdlib.h>
    7 |
codeN1109.c:15:9: warning: Array subscript is undefined
        ptr3[sub] = i;
        ^~~~~~~~
codeN1109.c:32:24: warning: Result of 'malloc' is converted to a pointer of type 'int', which is incompatible with sizeof operand type 'char'
    ptr_to_arr = (int*)malloc(sizeof(char));
                 ~~~~ ^~~~~~ ~~~~~~~~~~~~
codeN1109.c:37:5: warning: 2nd function call argument is an uninitialized value
    array_print(ptr_to_arr,size_array);
    ^~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
3 warnings generated.
scan-build: 3 bugs found.
scan-build: Run 'scan-view /home/diptendu/Desktop/project5/2023-04-07-154616-10064-1' to examine bug reports.
diptendu@diptendu:~/Desktop/project5$
```

Fixes
Include stdlib, malloc operand mismatch ptr_to_arr = (int*)malloc(sizeof(int));
int sub=0;

After fix

```
diptendu@diptendu:~/Desktop/project5$ scan-build -o . gcc codeN1109.c
scan-build: Using '/usr/lib/llvm-10/bin/clang' for static analysis
scan-build: Removing directory '/home/diptendu/Desktop/project5/2023-04-07-155513-10320-1' because it contains no reports.
scan-build: No bugs found.
diptendu@diptendu:~/Desktop/project5$
```

Other fixes
Added free for malloc
Null pointer check in array_print
Loop for 100000000 which is not defined by size_array passed as n, changing to n otherwise core dump

Final Fixed Code

```
// Use C-Lang Static Analyzer to analyze and fix the bugs
// Hint this code has bugs that can be identified using static analyzer
// Also review the output generated to find any bugs/vulnerabilities that the static analyzer misses.

//MACROS
#include<stdio.h>
// include stdlib for malloc
#include<stdlib.h>


//another custom array printing an array
```

```
void array_print(int* ptr3, int n)
{
      // initialize sub can be removed tbh
   int sub=0;

   // added null pointer check
   if (ptr3 == NULL) {
      printf("Error: Pointer is NULL.\n");
      return;
   }

   // passing n but not using n , this loop runs for 100000000 which is not defined by size_array
passed as n, changing to n, also this many allocation leads to core dump
   for(int i=0; i<n; i++)
   {
      ptr3[sub] = i;
      sub=sub+1;
      printf("Array is:%d\n ",ptr3[i]);
   }


}

int main(){
   // variable initializations
   int* ptr_to_arr;
   int size_array=10;

   //Note for developers(students): use variables in method using defined pointers (not directly)

   ptr_to_arr = (int*)malloc(sizeof(int));
   // Hint:: add and assign the references to the pointers here:


   //function call statements
   array_print(ptr_to_arr,size_array);

   // added free for malloc
   free(ptr_to_arr);
   return 0;


}
```

Scan build after fix

```
diptendu@diptendu:~/Desktop/project5$ scan-build -o . gcc codeN1109.c
scan-build: Using '/usr/lib/llvm-10/bin/clang' for static analysis
scan-build: Removing directory '/home/diptendu/Desktop/project5/2023-04-07-155801-10425-1' because it contains no reports.
scan-build: No bugs found.
diptendu@diptendu:~/Desktop/project5$ ./a.out
Array is:0
 Array is:1
 Array is:2
 Array is:3
 Array is:4
 Array is:5
 Array is:6
 Array is:7
Array is:8
 Array is:9
 diptendu@diptendu:~/Desktop/project5$
```

7. codeN1110.c

```
diptendu@diptendu:~/Desktop/project5$ scan-build -o . gcc codeN1110.c
scan-build: Using '/usr/lib/llvm-10/bin/clang' for static analysis
codeN1110.c:13:49: warning: Dereference of null pointer (loaded from variable 'ptr1')
    printf("Original First Number is : %d \n ", (*ptr1));
                                                ^~~~~~~
1 warning generated.
scan-build: 1 bug found.
scan-build: Run 'scan-view /home/diptendu/Desktop/project5/2023-04-07-160237-10515-1' to examine bug reports.
diptendu@diptendu:~/Desktop/project5$
```

Problems detected by clang and manual running

Variable first_number initialized - int first_number=9001;
Pointers ptr_to_first_number and ptr_to_char are not assigned to the address of first_number and choice, respectively.
int* ptr_to_first_number = &first_number;
char *ptr_to_char = &choice;
added nul pointer check in reverse_number
implemented reverse logic

Fixed Code

```
// Use C-Lang Static Analyzer to analyze and fix the bugs
// Also review the output generated to find any bugs that the static analyzer misses.
// Hint this code has 2-3 bugs that can be identified using static analyzer, 2 Tasks will have to be completed
//1.. add null pointer check condition for ptr1 in the reverse method for reversing logic.
//2.. Complete reversing Logic using given pointer.

//MACROS
#include<stdio.h>

// custom method to reverse a number (without using third variable/pointer) passed to it as arguments from main()
void reverse_number(int *ptr1, char* ptr2){
   int reverse = 0; //will be used as reversing variable.

   // adding null pointer check
   if (ptr1 == NULL || ptr2 == NULL) {
      printf("Error: Null pointer detected.\n");
      return;
   }

   printf("Original First Number is : %d \n ", (*ptr1));
   printf("Choice is : %c \n ", (*ptr2));

   // ---------------------- Secure Coding Task -------------------------////
   // Task 1: Implement a NULL pointer check condition (use if statment) (most important secure coding practice)

   // ------  add null pointer check condition for ptr1 Here-------

      // this second ptr2 != NULL is not needed anymore as check is already done above but still keeping it
   if(*ptr2 == 'Y' && ptr2 != NULL)
   {
      //Task 2 : complete logic for reversing a number (Hint: required statements 2)
      //----- add logic here ------
      while(*ptr1 > 0)
      {
         //add reversing logic here using ptr1 (which points to the first_number defined in main)
         // adding reverse logic
         reverse = reverse * 10 + (*ptr1 % 10);
         *ptr1 /= 10;
```

```
    }
    *ptr1=reverse;
    //print statements
    printf("Reversed First Number is : %d \n", (*ptr1));
    printf("Confirmed choice is : %c \n", (*ptr2));
  }
  else{
    printf("Sorry Not allowed\n");
  }

}

int main(){
  // variable initializations
  // number not initialized
  int first_number=9001;
  char choice = 'Y';

  //Note for developers(students): use variables in method using defined pointers (not directly)

    // pointers not set to inital values
  int* ptr_to_first_number = &first_number;
  char *ptr_to_char = &choice;
  // Hint:: add and assign the references to the pointers here:


  //function call statements
  reverse_number(ptr_to_first_number, ptr_to_char);

  return 0;

}
```

Scan build after fix

```
diptendu@diptendu:~/Desktop/project5$ scan-build -o . gcc codeN1110.c
scan-build: Using '/usr/lib/llvm-10/bin/clang' for static analysis
scan-build: Removing directory '/home/diptendu/Desktop/project5/2023-04-07-161705-11110-1' because it contains no reports.
scan-build: No bugs found.
diptendu@diptendu:~/Desktop/project5$ ./a.out
Original First Number is : 9001
 Choice is : Y
 Reversed First Number is : 1009
Confirmed choice is : Y
diptendu@diptendu:~/Desktop/project5$
```

8. codeN1111.c

```
diptendu@diptendu:~/Desktop/project5$ scan-build -o . gcc codeN1111.c
scan-build: Using '/usr/lib/llvm-10/bin/clang' for static analysis
codeN1111.c:21:1: warning: Potential leak of memory pointed to by 'square'
}
^
1 warning generated.
scan-build: 1 bug found.
scan-build: Run 'scan-view /home/diptendu/Desktop/project5/2023-04-07-162114-11150-1' to examine bug reports.
diptendu@diptendu:~/Desktop/project5$
```

This is due to calloc not freed after printing matrix
Fix code available in Fix

Clang output now

```
diptendu@diptendu:~/Desktop/project5$ scan-build -o . gcc codeN1111.c
scan-build: Using '/usr/lib/llvm-10/bin/clang' for static analysis
scan-build: Removing directory '/home/diptendu/Desktop/project5/2023-04-07-162626-11344-1' because it contains no reports.
scan-build: No bugs found.
diptendu@diptendu:~/Desktop/project5$
```

Additional problem

```
if (matrix_size > MAX_LEN) {
      printf("Exceeded max length!");
   }
```

When input is greater than defined MAX_LEN of 10 , it just prints a message but still actually builds and displays the matrix, so essentially the LEN check is not valid as I can provide input of 10000

Added an else case, also can be returned inside the if check.

```
   if (matrix_size > MAX_LEN) {
      printf("Exceeded max length!");
   }
   else {
   buildMatrix(matrix_size);
     }
   return 0;
```

Also, scanf can be made better

```
int ret = scanf("%d", &matrix_size);

   if (ret != 1) {
      printf("Invalid input!\n");
      return 1;
   }
```

Fixed code

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

#define MAX_LEN 10

void buildMatrix(unsigned int matrix_size) {
   char **square = calloc(matrix_size, sizeof(char *));
   for (int i = 0; i < matrix_size; ++i) {
      square[i] = calloc(matrix_size, sizeof(char));
   }
   system("cls");
   printf("\tMatrix\n");

   for (int i = 0; i < matrix_size; i++) {
      printf("\n");
      for (int j = 0; j < matrix_size; j++) {
         square[i][j] = i + j;
         printf("  %d  |", i + j);
      }
   }

   // added code for free
     for (int i = 0; i < matrix_size; ++i) {
     free(square[i]);
   }
   free(square);

}

int main() {
   int matrix_size = 3;
   printf("Enter size of n x n matrix");

   // additional check for scanf cos you can input alphabets


   int ret = scanf("%d", &matrix_size);

   if (ret != 1) {
      printf("Invalid input!\n");
      return 1;
   }
```

```
    if (matrix_size > MAX_LEN) {
       printf("Exceeded max length!");
    }

    // if within max len then build matrix
    else {
    buildMatrix(matrix_size);
      }
    return 0;

}
```

Scan-build after fix

```
diptendu@diptendu:~/Desktop/project5$ scan-build -o . gcc codeN1111.c
scan-build: Using '/usr/lib/llvm-10/bin/clang' for static analysis
scan-build: Removing directory '/home/diptendu/Desktop/project5/2023-04-07-170706-11715-1' because it contains no reports.
scan-build: No bugs found.
diptendu@diptendu:~/Desktop/project5$ ./a.out
Enter size of n x n matrix101
Exceeded max length!diptendu@diptendu:~/Desktop/project5$ ./a.out
Enter size of n x n matrix5
sh: 1: cls: not found
        Matrix

  0  |  1  |  2  |  3  |  4  |
  1  |  2  |  3  |  4  |  5  |
  2  |  3  |  4  |  5  |  6  |
  3  |  4  |  5  |  6  |  7  |
  4  |  5  |  6  |  7  |  8  |diptendu@diptendu:~/Desktop/project5$ ./a.out
Enter size of n x n matrixabcd
Invalid input!
diptendu@diptendu:~/Desktop/project5$
```

9. codeN1112.c

```
diptendu@diptendu:~/Desktop/project5$ scan-build -o . gcc codeN1112.c
scan-build: Using '/usr/lib/llvm-10/bin/clang' for static analysis
codeN1112.c:10:9: warning: Branch condition evaluates to a garbage value
    if (px) {
        ^~
1 warning generated.
scan-build: 1 bug found.
scan-build: Run 'scan-view /home/diptendu/Desktop/project5/2023-04-07-171744-11841-1' to examine bug reports.
diptendu@diptendu:~/Desktop/project5$
```

Fixes

int *px = (int *) malloc(sizeof(int));

clang output now

```
diptendu@diptendu:~/Desktop/project5$ scan-build -o . gcc codeN1112.c
scan-build: Using '/usr/lib/llvm-10/bin/clang' for static analysis
codeN1112.c:19:9: warning: Attempt to free released memory
        free(px);
        ^~~~~~~~
1 warning generated.
scan-build: 1 bug found.
scan-build: Run 'scan-view /home/diptendu/Desktop/project5/2023-04-07-171850-11952-1' to examine bug reports.
diptendu@diptendu:~/Desktop/project5$
```

This is because an additional fix in the else clause, removing that such that the outer free is always reached. (double free)

Fix

Remove the free in else

```
diptendu@diptendu:~/Desktop/project5$ scan-build -o . gcc codeN1112.c
scan-build: Using '/usr/lib/llvm-10/bin/clang' for static analysis
scan-build: Removing directory '/home/diptendu/Desktop/project5/2023-04-07-172102-12060-1' because it contains no reports.
scan-build: No bugs found.
diptendu@diptendu:~/Desktop/project5$ ./a.out
   infdiptendu@diptendu:~/Desktop/project5$
```

Additional problems

Divide by 0, the line *px = argc - 1; means if argc =1 (program name only) this will result in division by 0. Either remove the argc which is not needed as we are not doing anything with arguments or add additional check which prevents division by 0.

Fixed code

// Fix the unintialized variable using appropriate technique that fits according to the rest of the code

```
#include <stdio.h>

#include <stdlib.h>


int main(int argc, char **argv) {


        // initialize variable
  int *px = (int *) malloc(sizeof(int));
  float foo;


  if (px) {
    foo = 3.5;
    *px = argc - 1;
```

```
    if (*px == 1) {
        printf("%6.1f", foo);
    } else {
                            // add check for div by 0


                            if (*px != 0) {
        printf("%6.1f", 100.00 / *px);
        } else {
            printf("Division by zero is not allowed. Add aruguments\n");
        }


        //free(px); removed this as outer free handles this - double free
    }
    free(px);
  }
  return 0;
}
```

Scan build after fix

Part 3: Mapping Bug(s) identified to categories in CWE Database

1. codeN1104.c
   CWE-457: Use of Uninitialized Variable and CWE-456: Missing Initialization of a Variable
   Since the variable maxRevenueTeam was not initialized
   CWE-704: Incorrect Type Conversion or Cast
   Since we were casting long to short in calculation
2. codeN1105.c
   CWE-476: NULL Pointer Dereference
   Since username and password checks were missing
   CWE-758: Reliance on Undefined, Unspecified, or Implementation-Defined Behavior
   strcmp without null check leads to this
3. codeN1106.c
   CWE-476: NULL Pointer Dereference
   Since int *ptr_to_first_number = &first_number; and int *ptr_to_size = &size;
   were set to NULL
   CWE-457: Use of Uninitialized Variable
   in void method – count was not initialized.
   CWE-840: Business Logic Errors (4.10)
   ARRAY_MEMORY_SIZE is 10 but in the method, the loop is till 30.
4. codeN1107.c
   CWE-476: NULL Pointer Dereference
   Since int *ptr_to_first_number = &first_number; and
   int *ptr_to_second_number = &second_number; were set to NULL.
   CWE-840: Business Logic Errors (4.10) - Swapping logic missing
5. codeN1108.c
   CWE-476: NULL Pointer Dereference
   Since int *ptr_to_first_number = &first_number;
   and char *ptr_to_char = &choice; were NULL
   CWE-457: Use of Uninitialized Variable (4.10)
   Since int first_number=10; and int size_array=10; were not set
   CWE-401: Missing Release of Memory after Effective Lifetime
   missing free at the end
6. codeN1109.c
   CWE-457: Use of Uninitialized Variable (4.10)
   Since int sub=0; was not set
   CWE-401: Missing Release of Memory after Effective Lifetime
   missing free at the end
   CWE-840: Business Logic Errors (4.10)

Loop for 100000000 which is not defined by size_array passed as n, changing to n otherwise core dump

7. codeN1110.c
   CWE-457: Use of Uninitialized Variable (4.10)
   int first_number=9001; was not set
   CWE-476: NULL Pointer Dereference
   int* ptr_to_first_number = &first_number; and char *ptr_to_char = &choice; were null
   CWE-840: Business Logic Errors (4.10)
   Reversing logic missing
8. codeN1111.c
   CWE-762: Mismatched Memory Management Routines
   CWE-401: Missing Release of Memory after Effective Lifetime
   due to `square` not freed
   CWE-840: Business Logic Errors (4.10)
   matrix_size > MAX_LEN was not preventing the matrix building just printing an error message
9. codeN1112.c
   CWE-457: Use of Uninitialized Variable (4.10)
   since int *px = (int *) malloc(sizeof(int)); was not initialized
   CWE-415: Double Free
   in the else case ther is an additional free
   CWE-369: Divide By Zero
   in the case where there is 1 argument

Part 4: Sanitizing Bugs and Secure programs automatically

Followed the installation procedure.

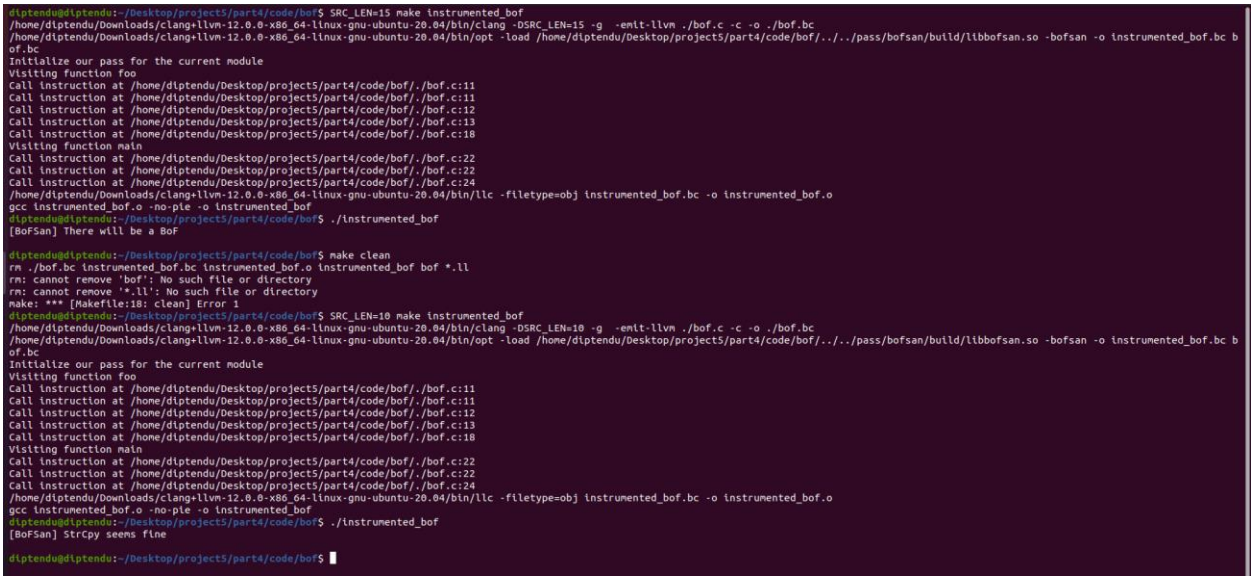
Edited the make file according to README.

Added following lines to the bofsan.cpp

Uncommented

```
// get the lengths
        CallInst* dest_len = builder.CreateCall(StrlenFunc, dest_str);
        CallInst* src_len = builder.CreateCall(StrlenFunc, src_str);
Instruction *len_check = cast<Instruction>(builder.CreateICmpSGT(src_len, dest_len, "len_check"));
// ok
        builder.CreateCall(PutsFunc, {strPtr1});
  // not ok and exit
         builder.CreateCall(PutsFunc, {strPtr2});
                    builder.CreateCall(ExitFunc, {ConstantInt::get(Type::getInt32Ty(F.getContext()), 1)});
```

Test screenshot



GITHUB URL - https://github.com/diptendukar/project5