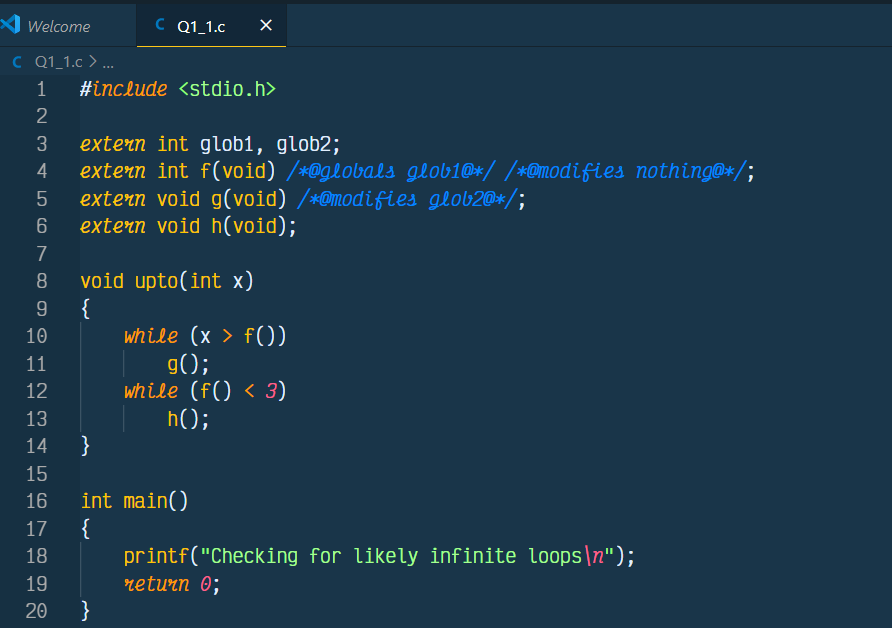
Assignment 3

Submitted by U17CO110

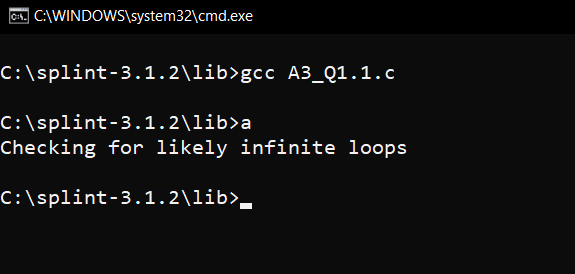
Question 1:

1. Infinite Loops

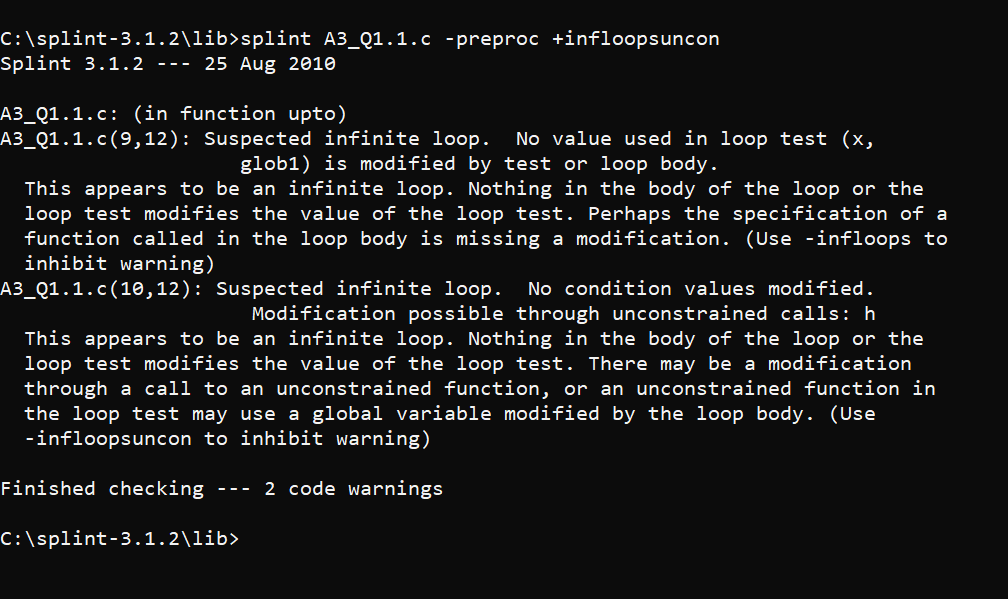
Program



GCC Output



Splint Output

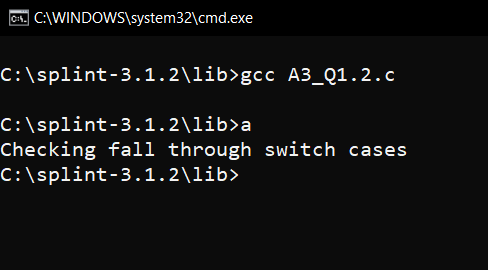


1. Fall through switch cases

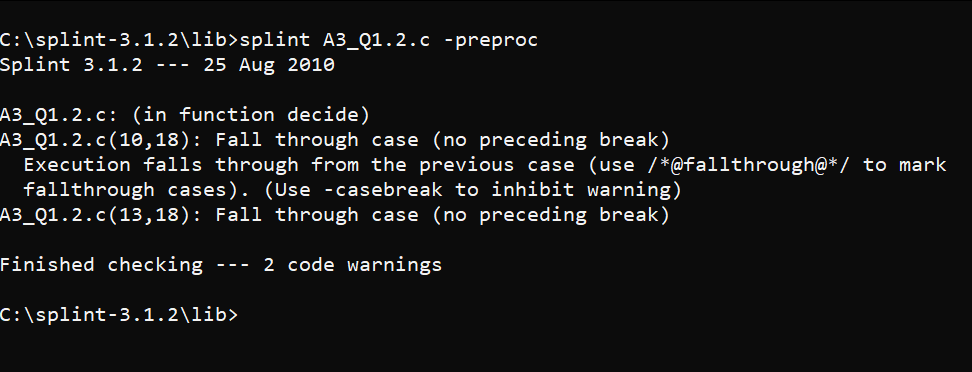
Program



GCC Output



Splint Output



1. Missing switch case

Program

*#include* <stdio.h>

*typedef* enum { YES, NO, DEFINITELY, PROBABLY, MAYBE } ynm;

void decide(ynm y)

{

*switch* (y)

    {

*case* PROBABLY:

*break*;

*case* NO:

        printf("No!");

*break*;

*case* MAYBE:

        printf("Maybe");

*break*;

*/\*@fallthrough@\*/*

*case* YES:

        printf("Yes!");

*break*;

    }

}

int main()

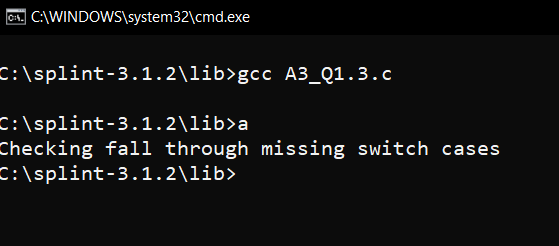
{

    printf("Checking fall through missing switch cases");

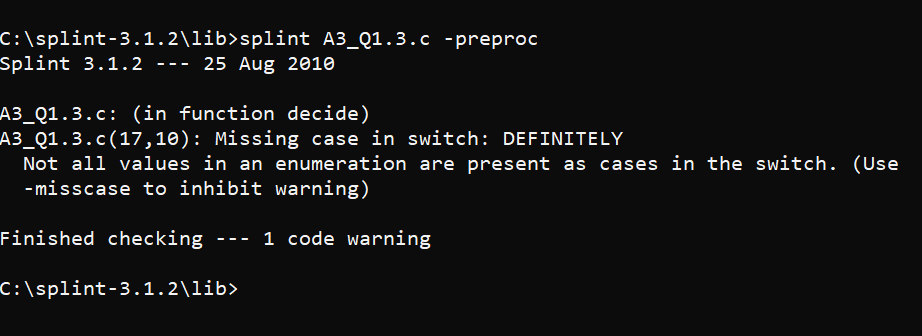
*return* *0*;

}

GCC Output

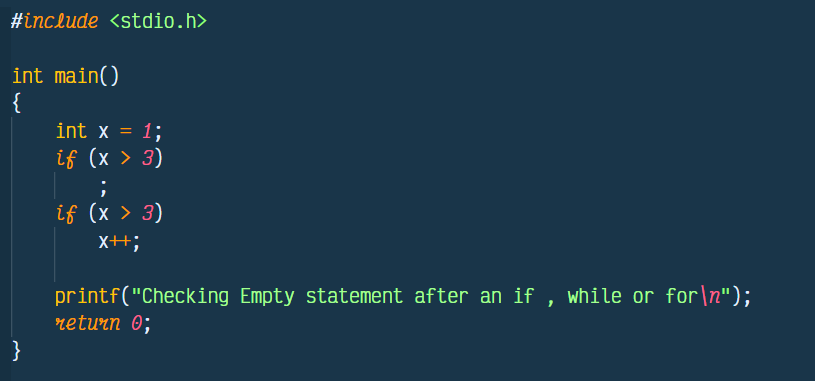


Splint Output

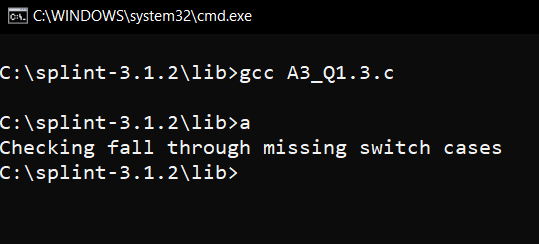


1. Empty statement after an if, while or for statement

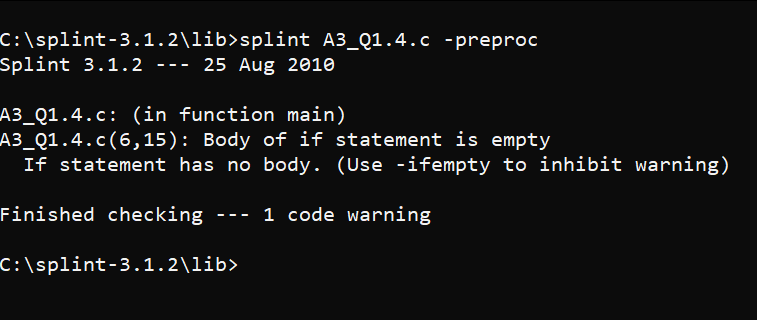
Program



GCC Output



Splint Output



Question 2:

Buffers are memory storage regions that temporarily hold data while it is being transferred from one location to another. A buffer overflow (or buffer overrun) occurs when the volume of data exceeds the storage capacity of the memory buffer. As a result, the program attempting to write the data to the buffer overwrites adjacent memory locations

Program

*#include* <stdio.h>

void updateEnv(char \*str)

{

    char \*tmp;

    tmp = getenv("MYENV");

*if* (tmp != *NULL*)

        strcpy(str, tmp);

}

void updateEnvSafe(char \*str, size\_t strSize) */\*@requires maxSet(str @ A3\_Q2.c(18,9)) >= strSize @ A3\_Q2.c(18,13)@\*/*

{

    char \*tmp;

    tmp = getenv("MYENV");

*if* (tmp != *NULL*)

    {

        strncpy(str, tmp,

                strSize - *1*);

        str[strSize - *1*] = '/0';

    }

}

int main()

{

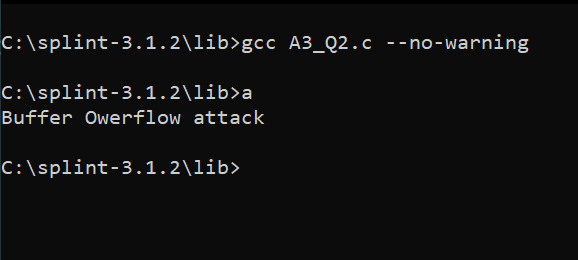
    printf("Buffer Owerflow attack*\n*");

*return* *0*;

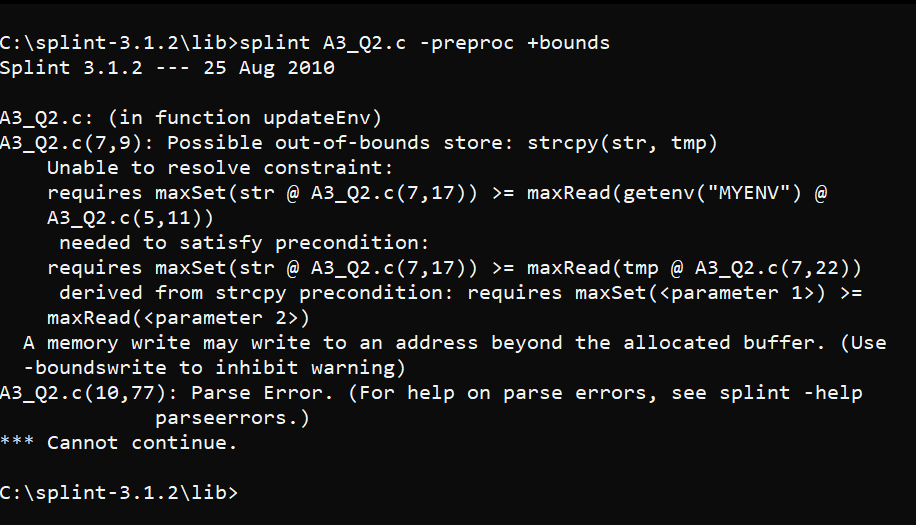
}

The above program shows two ways of updating the environment variables. The first function doesn’t update it safely and can lead to buffer overflow attack whereas the second function is the correct method to update the environment variable and doesn’t cause any buffer overflow attack.

GCC Output



Splint Output



Question 3:

Program

*#include* <stdio.h>

*extern* int square(*/\*@sef@\*/* int x);

*#define* square(x) ((x) \* (x))

*extern* int sumsquares(int x, int y);

*#define* sumsquares(x, y) (square(x) + square(y))

int main()

{

    int i = *1*;

    i = square(i++);

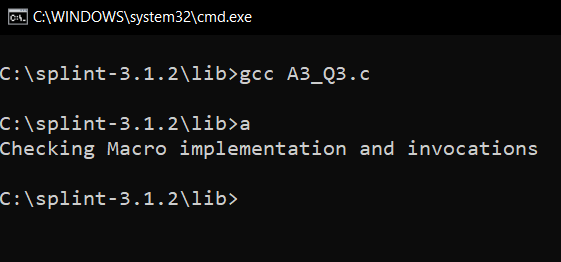
    i = sumsquares(i, i);

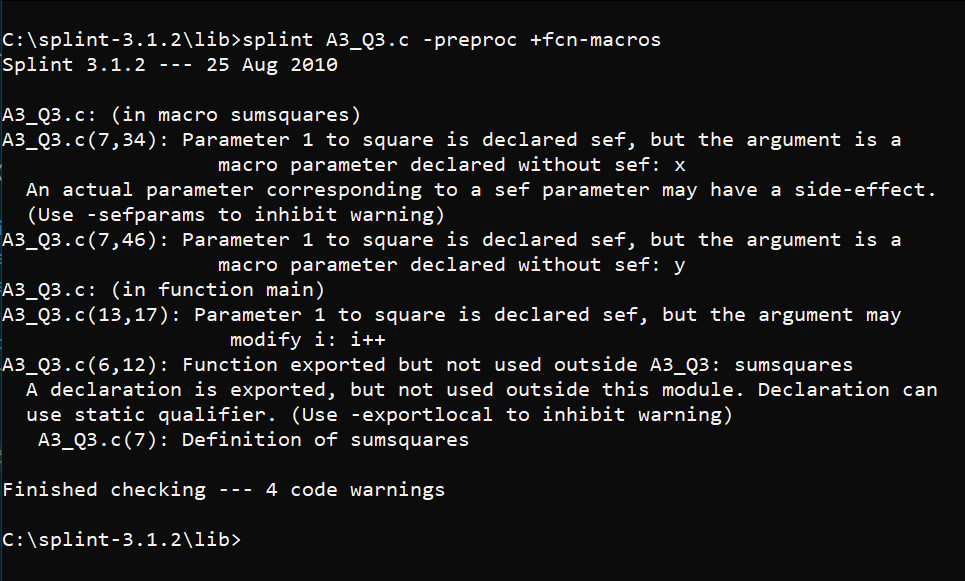
    printf("Checking Macro implementation and invocations*\n*");

*return* *0*;

}

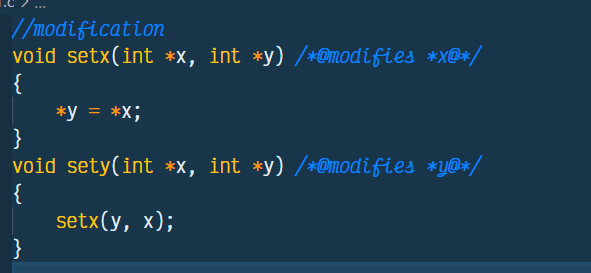
GCC Output



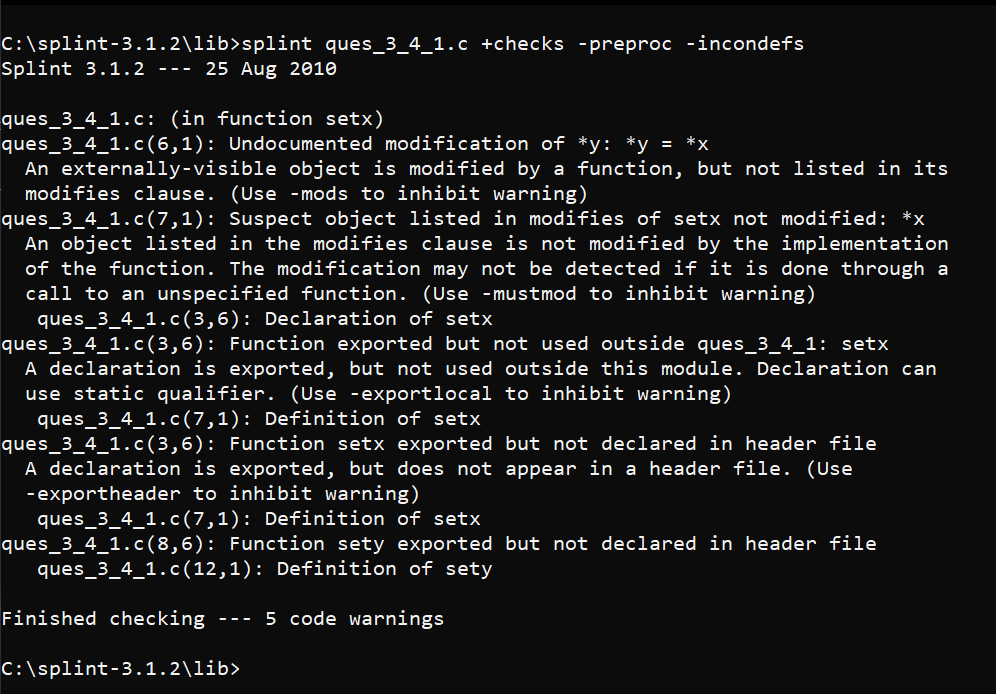
Splint Output

Question 4:

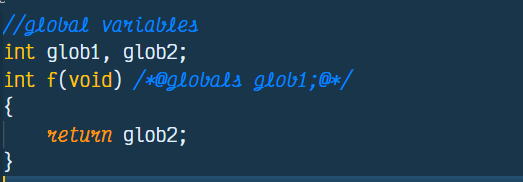
Program 1:



Splint Output



Program 2:



Splint Output

