1. Overview: Programmers: Yves Chen (ychen22) and Max Peicher (mpeich01) Bomb number: 26 Ackowledgements: Ben London, Josh Simani, Daniel Jelcic, Ryan Polhemus Hour approximations: 21 hours
2. DEFUSE inputs: I have no reservations whatsoever about holding him back another year. 10 15 20 25 30 35 1 270 16 opekma 526
3. Descrption Phase 1: takes an input string and calls strings not equal to compare a constant string stored at 0x401c50 if it is the same string, it defuses the phase, else it explodes.

Phase 2: this phase requires 6 numbers, where each number is incremented by a value of 5 from the previous (ie 5, 10, 15, etc.) It does this by adding 5 to the current number and comparing it to the next. If they are not the same, the bomb explodes. If they are, the phase is solved.If you enter anything thats not 6 numbers, the bomb explodes.

phase 3: this phase has a switch statment and requires 2 integer inputs from the user. If the user inputs 1, it jumps to 0x401109 within the switch statement. This will then compare the second user input to a hard coded value of 0x10e, which is 270 in decimal. If they are not the same, the bomb explodes. If they are, the phase is solved. If the user's first input was a different number the switch statement will jump to a different conditional, and compare with a different hardcoded value. If they are the same, the phase will be defused. If they are different the bomb blows up.

phase 4: this phase calls func4, which is a recursive fibonacci call. func4 takes the user input(n) ,and does the fibonacci sequence n times. It then returns this value. In phase_4, that value is compared to 1597. If they are not equal, the bomb explodes. if they are, the phase is solved.

4. Code: phase 5 & 6:

/** * test.c * Assignment: Homework 5 for Comp 40, Fall 2019 * Authors: Max Peicher (mpeich01) and Yves Chen (ychen22) * Date: November 6, 2019 Summary: This program compresses and decompresses ppm images **/

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

/* constant string stored as address in phase 5 / *const char* GIANTS = "giants";

/* mock linkedlist node used in phase 6 / *typedef struct node{ int data; struct node* next; } Node;

void phase_5(char * input); void phase_6(char * input);

extern void explode_bomb(); extern int string_length(char * input); extern bool string_not_equal();

/* helper function used by phase 6 that sorts a linked list from greatest to least / extern Node fun6(Node *head);

/ purpose: This phase takes the input of the user, which is a a string of 6 chars. If the string length != 6, it calls explode_bomb. It then goes through each individual char of the string, and performs a a logical and with that letter, then storing the new value in the place of the original. This will then change our word. after all of the letters have gone through this, the program then compares the new word to a const string, which for us was “giants”. If they are the not the same, it calls explode bomb. / void phase_5(char * input){

if (string_length(input) != 6) /* checks to make sure string size = 6 / explode_bomb(); else{ for (int i = 0; i < 6; i++){ / for each letter in string / input[i] = (input[i] & 15);// perform logical & with curr letter and 15 } // this result will be stored in “input” if (string_not_equal(input, GIANTS)) / if new string != expected const */ explode_bomb(); } return; }

/ purpose: Uses linkedlist of hard coded values with 9 elements. The function then calls fun6 which sorts the list in greatest to least order. Then the function traverses the newly sorted list to the third largest element and compares that with the user input. If they are the same value, then the phase is defused, else it calls explode_bomb. / void phase_6(char * input){

/ This phase uses a pre-established linked list, which has 9 nodes, whose values are hard coded, and are in the following order: 526,520,307,854,295,941,50,208,526,419 For compilation purposes, we have made our own Node type and declared an instance within phase_6. It is to be assumed to our list variable contains the first node of the aforementioned linked list. / Node *list;

/* fun6 sorts the list of values from greatest to least */ list = fun6(list);

int inputNum; int listValue;

/* Traverses list to third element and stores that value in listValue var/ list = list -> next; list = list -> next; listValue = list -> data; / This value is the third largest value of the linked list newly sorted by func6 */

/* converts user input to integer*/ inputNum = atoi(input);

/* compares listValue with user input and returns if the same integer, if not equal it explodes the bomb */ if (listValue == inputNum){ return; } else explode_bomb();

}