Output of running f1()

A screenshot of a computer

Description automatically generated

Output of running f2();

A screenshot of a computer program

Description automatically generated

Output of running f3();

A screenshot of a computer program

Description automatically generated

Code for project 4

#include <stdio.h>

#include <stdlib.h>

// macro defition for array size

#define SIZE 1000

// function definitions

void f1();

void f2();

void f3();

int main () {

// f1();

// f2();

f3();

return 0;

}

void f1() {

// define local array of char in which size can be adjustable using define.

char charArray[SIZE];

// increment by 1 when each next function call is made

static int n = 1;

// stores starting address of the array

static long int addr;

// print output

printf("Call #%d \tat %p\n", n, charArray);

printf("AR size\t #%d\t is %ld\n", n, addr - (long)(charArray));

n++;

addr = (long)charArray;

if (n <= 10) {

f1();

} else {

return;

}

}

void f2() {

// define local array of char in which size can be adjustable using define.

char charArray[SIZE];

// increment by 1 when each next function call is made

static int n;

// stores starting address of the array

static long int addr;

// print output

printf("Call #%d \tat %p\n", n, charArray);

printf("AR size\t #%d\t is %ld\n", n, addr - (long)(charArray));

printf("Stack size\t #%d\t is %ld\n", n, n\*addr);

n++;

addr = (long)charArray;

f2();

}

void f3() {

// define local array of char in which size can be adjustable using define and we use malloc for dynamic.

char\* charArray = (char\*)malloc(sizeof(char)\*SIZE);

// increment by 1 when each next function call is made

static int n = 1;

// stores starting address of the local variable c now

static long int addr;

// used to calculate size of AR since array won't be used

char c = 'a';

// print output

printf("Call #%d \tat %p\n", n, &c);

printf("AR size\t #%d\t is %ld\n", n, addr - (long)(&c));

n++;

addr = (long)&c;

free(charArray);

if (n <= 10) {

f3();

} else {

return;

}

}