

Name: Kakaraparty Srirama Srikar
Roll Number: 142301013
Lab: LAB 1 C PROGRAMS

```
C 1_hello.c > main(void)
1  #include <stdio.h>
2
3  int main(void) {
4      printf("Hello, World!\n");
5      return 0;
6  }
7

PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS

[student@nil-316-052l lab1_cprograms]$ gcc 1_hello.c -o hello
[student@nil-316-052l lab1_cprograms]$ ./hello
Hello, World!
[student@nil-316-052l lab1_cprograms]$
```

The above is the image of the program and output of Program 2.1, which is a program that prints "Hello, World!"

```
C 2_simple_io.c > ...
1  #include <stdio.h>
2
3  int main(void){
4      int number;
5      printf("Enter an integer: ");
6      scanf("%d", &number);
7      printf("You entered: %d\n", number);
8      return 0;
9  }
10
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

- [student@nil-316-052l lab1_cprograms]\$ gcc 2_simple_io.c -o simple_io
- [student@nil-316-052l lab1_cprograms]\$./simple_io
Enter an integer: 5335509
You entered: 5335509
- [student@nil-316-052l lab1_cprograms]\$ █

The above is the image of the program and output of Program 2.2, which is a program that takes a number as an input and then prints it...

```
C 3_control_flow.c > ...
1  #include <stdio.h>
2
3  int main(void) {
4      int i, sum = 0;
5      for (i = 1; i <= 5; i++) {
6          if (i % 2 == 0) {
7              printf("%d is even\n", i);
8          } else {
9              printf("%d is odd\n", i);
10         }
11         sum += i;
12     }
13
14     printf("Sum of numbers 1 to 5 is %d\n", sum);
15     return 0;
16 }
17
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
● [student@nil-316-052l lab1_cprograms]$ gcc 3_control_flow.c -o control_flow
● [student@nil-316-052l lab1_cprograms]$ ./control_flow
1 is odd
2 is even
3 is odd
4 is even
5 is odd
Sum of numbers 1 to 5 is 15
○ [student@nil-316-052l lab1_cprograms]$
```

The above is the image of the program and output of Program 2.3.

```
C 4_array_and_summation.c > main(void)
1  #include <stdio.h>
2
3  #define SIZE 5
4
5  int main(void) {
6      int arr[SIZE] = {1, 2, 3, 4, 5};
7      int sum = 0;
8
9      for (int i = 0; i < SIZE; i++) {
10         sum += arr[i];
11     }
12
13     printf("Sum of array elements = %d\n", sum);
14     return 0;
15 }
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

- [student@nil-316-052l lab1_cprograms]\$ gcc 4_array_and_summation.c -o array_and_sum
- [student@nil-316-052l lab1_cprograms]\$./array_and_sum
Sum of array elements = 15
- [student@nil-316-052l lab1_cprograms]\$

The above is the image of the program and output of Program 2.4.

```
C 5_pointer_basics.c > main(void)
1  #include <stdio.h>
2
3  int main(void) {
4      int x = 10;
5      int *ptr = &x; // pointer to x
6
7      printf("Value of x: %d\n", x);
8      printf("Address of x: %p\n", (void*)&x);
9      printf("Value of ptr: %p\n", (void*)ptr);
10     printf("Value pointed by ptr: %d\n", *ptr);
11
12     return 0;
13 }
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
• [student@nil-316-052l lab1_cprograms]$ gcc 5_pointer_basics.c -o pointer_basics
• [student@nil-316-052l lab1_cprograms]$ ./pointer_basics
Value of x: 10
Address of x: 0x7ffe435fe914
Value of ptr: 0x7ffe435fe914
Value pointed by ptr: 10
○ [student@nil-316-052l lab1_cprograms]$
```

The above is the image of the program and output of Program 2.5.

```
C 6_pointer_arithmetic.c > ...
1  #include <stdio.h>
2
3  int main(void) {
4      int arr[3] = {10, 20, 30};
5      int *p = arr; // points to arr[0]
6
7      printf("p points to arr[0]: value = %d\n", *p);
8      p++; // Move to arr[1]
9      printf("p now points to arr[1]: value = %d\n", *p);
10     p++; // Move to arr[2]
11     printf("p now points to arr[2]: value = %d\n", *p);
12
13     return 0;
14 }
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
• [student@nil-316-052l lab1_cprograms]$ gcc 6_pointer_arithmetic.c -o pointer_arithmetic
• [student@nil-316-052l lab1_cprograms]$ ./pointer_arithmetic
p points to arr[0]: value = 10
p now points to arr[1]: value = 20
p now points to arr[2]: value = 30
○ [student@nil-316-052l lab1_cprograms]$
```

The above is the image of the program and output of Program 2.6.

```
C 7.swap.c > main(void)
1  #include <stdio.h>
2
3  void swap(int *a, int *b) {
4      int temp = *a;
5      *a = *b;
6      *b = temp;
7  }
8
9  int main(void) {
10     int x = 5, y = 10;
11     printf("Before swap: x = %d, y = %d\n", x, y);
12     swap(&x, &y);
13     printf("After swap: x = %d, y = %d\n", x, y);
14     return 0;
15 }
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
sh-5.2$ gcc 7.swap.c -o swap
sh-5.2$ ./swap
Before swap: x = 5, y = 10
After swap: x = 10, y = 5
sh-5.2$ □
```

The above is the image of the program and output of Program 2.7.

```
C 8_dynamic_memory_allocation.c > main(void)
1  #include<stdio.h>
2  #include<stdlib.h>
3
4  int main(void){
5      int n;
6      printf("Enter the number of elements: ");
7      scanf("%d", &n);
8
9      int *arr = (int *)malloc(n*sizeof(int));
10     if(arr==NULL){
11         printf("Memory allocation has failed!\n");
12         return 1;
13     }
14     for (int i = 0; i < n; i++) {
15         arr[i] = i + 1;
16     }
17
18     // Print values
19     printf("Allocated array elements:\n");
20     for (int i = 0; i < n; i++) {
21         printf("%d ", arr[i]);
22     }
23     printf("\n");
24
25     free(arr); // Release the allocated memory
26     return 0;
27
28 }
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS sh + v

```
sh-5.2$ gcc 8_dynamic_memory_allocation.c -o dynamic_memory
sh-5.2$ ./dynamic_memory
Enter the number of elements: 20
Allocated array elements:
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20
sh-5.2$
```

The above is the image of the program and output of Program 2.8.


```
C 9_sturcture_basics.c > main(void)
1  #include <stdio.h>
2  #include <string.h>
3
4  struct Student {
5      char name[50];
6      int age;
7  };
8
9  int main(void) {
10     struct Student s1;
11     strcpy(s1.name, "Alice");
12     s1.age = 20;
13     printf("Student Name: %s, Age: %d\n", s1.name, s1.age);
14     return 0;
15 }
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS sh + v [] []

```
sh-5.2$ gcc 9_sturcture_basics.c -o struct_basics
sh-5.2$ ./struct_basics
Student Name: Alice, Age: 20
sh-5.2$ [ ]
```

The above is the image of the program and output of Program 2.9.

```

C 10_linked_list.c > main(void)
1  #include <stdio.h>
2  #include <stdlib.h>
3
4  struct Node {
5      int data;
6      struct Node *next;
7  };
8
9  void insertAtHead(struct Node **head, int value) {
10     struct Node *newNode = (struct Node*)malloc(sizeof(struct Node));
11     newNode->data = value;
12     newNode->next = *head;
13     *head = newNode;
14 }
15
16 void printList(struct Node *head) {
17     struct Node *current = head;
18     while (current != NULL) {
19         printf("%d -> ", current->data);
20         current = current->next;
21     }
22     printf("NULL\n");
23 }
24
25 int main(void) {
26     struct Node *head = NULL;
27
28     insertAtHead(&head, 30);
29     insertAtHead(&head, 20);
30     insertAtHead(&head, 10);
31
32     printList(head);
33     return 0;
34 }

```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```

sh-5.2$ gcc 10_linked_list.c -o linked_list
sh-5.2$ ./linked_list
10 -> 20 -> 30 -> NULL
sh-5.2$

```

The above is the image of the program and output of Program 2.10.

```

C 11_stack.c > isFull(Stack *)
1  #include <stdio.h>
2  #include <stdlib.h>
3  #define MAX_SIZE 5
4  typedef struct {
5      int top;
6      int arr[MAX_SIZE];
7  } Stack;
8  void initStack(Stack *s) {s->top = -1;}
9
10 int isFull(Stack *s) {return s->top == MAX_SIZE - 1;}
11
12 int isEmpty(Stack *s) {return s->top == -1;}
13
14 void push(Stack *s, int value) {
15     if (isFull(s)) {
16         printf("Stack overflow!\n");
17         return;
18     }
19     s->arr[++(s->top)] = value;
20 }
21
22 int pop(Stack *s) {
23     if (isEmpty(s)) {
24         printf("Stack underflow!\n");
25         return -1;
26     }
27     return s->arr[(s->top)--];
28 }
29
30 int main(void) {
31     Stack myStack;
32     initStack(&myStack);
33     push(&myStack, 10);
34     push(&myStack, 20);
35     push(&myStack, 30);
36     printf("Popped: %d\n", pop(&myStack));
37     printf("Popped: %d\n", pop(&myStack));
38     return 0;
39 }

```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```

sh-5.2$ gcc 11_stack.c -o stack
sh-5.2$ ./stack
Popped: 30
Popped: 20
sh-5.2$

```

The above is the image of the program and output of Program 2.11.

```

C 12_queue.c > main(void)
1  #include <stdio.h>
2  #include <stdlib.h>
3  #define MAX_SIZE 5
4
5  typedef struct {
6      int front, rear;
7      int arr[MAX_SIZE];
8  } Queue;
9
10 void initQueue(Queue *q) {
11     q->front = -1;
12     q->rear = -1;
13 }
14
15 int isEmpty(Queue *q) {
16     return q->front == -1;
17 }
18
19 int isFull(Queue *q) {
20     return (q->rear + 1) % MAX_SIZE == q->front;
21 }
22
23 void enqueue(Queue *q, int value) {
24     if (isFull(q)) {
25         printf("Queue is full!\n");
26         return;
27     }
28     if (q->front == -1)
29         q->front = 0;
30     q->rear = (q->rear + 1) % MAX_SIZE;
31     q->arr[q->rear] = value;
32 }
33

```

(cont'd)

```
C 12_queue.c > main(void)
34 int dequeue(Queue *q) {
35     if (isEmpty(q)) {
36         printf("Queue is empty!\n");
37         return -1;
38     }
39     int result = q->arr[q->front];
40     if (q->front == q->rear) {
41         // Only one element
42         q->front = -1;
43         q->rear = -1;
44     } else {
45         q->front = (q->front + 1) % MAX_SIZE;
46     }
47     return result;
48 }
49
50 int main(void) {
51     Queue myQueue;
52     initQueue(&myQueue);
53
54     enqueue(&myQueue, 1);
55     enqueue(&myQueue, 2);
56     enqueue(&myQueue, 3);
57     printf("Dequeued: %d\n", dequeue(&myQueue));
58     printf("Dequeued: %d\n", dequeue(&myQueue));
59     return 0;
60 }
61
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
sh-5.2$ gcc 12_queue.c -o queue
sh-5.2$ ./queue
Dequeued: 1
Dequeued: 2
sh-5.2$
```

The above is the images of the program and output of Program 2.12.

```

C 13_bubble_sort.c > main(void)
1  #include <stdio.h>
2
3  void bubbleSort(int arr[], int n) {
4      for (int i = 0; i < n - 1; i++) {
5          for (int j = 0; j < n - i - 1; j++) {
6              if (arr[j] > arr[j + 1]) {
7                  // Swap
8                  int temp = arr[j];
9                  arr[j] = arr[j + 1];
10                 arr[j + 1] = temp;
11             }
12         }
13     }
14 }
15
16 int main(void) {
17     int arr[] = {64, 34, 25, 12, 22, 11, 90};
18     int n = sizeof(arr) / sizeof(arr[0]);
19
20     printf("Original array:\n");
21     for (int i = 0; i < n; i++) {
22         printf("%d ", arr[i]);
23     }
24     printf("\n");
25
26     bubbleSort(arr, n);
27
28     printf("Sorted array:\n");
29     for (int i = 0; i < n; i++) {
30         printf("%d ", arr[i]);
31     }
32     printf("\n");
33
34     return 0;
35 }
36

```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```

sh-5.2$ gcc 13_bubble_sort.c -o bubble_sort
sh-5.2$ ./bubble_sort
Original array:
64 34 25 12 22 11 90
Sorted array:
11 12 22 25 34 64 90
sh-5.2$ 

```

The above is the image of the program and output of Program 2.13.


```
C 14_binary_search.c > main(void)
1  #include <stdio.h>
2
3  int binarySearch(int arr[], int size, int target) {
4      int low = 0;
5      int high = size - 1;
6
7      while (low <= high) {
8          int mid = (low + high) / 2;
9          if (arr[mid] == target) {
10             return mid;
11         }
12         else if (arr[mid] < target) {
13             low = mid + 1;
14         }
15         else {
16             high = mid - 1;
17         }
18     }
19
20     return -1;
21 }
22
23 int main(void) {
24     int arr[] = {2, 4, 6, 8, 10, 12};
25     int size = sizeof(arr) / sizeof(arr[0]);
26     int target = 8;
27     int result = binarySearch(arr, size, target);
28
29     if (result != -1) {
30         printf("Element %d found at index %d.\n", target, result);
31     } else {
32         printf("Element %d not found.\n", target);
33     }
34
35     return 0;
36 }
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
sh-5.2$ gcc 14_binary_search.c -o binary_search
sh-5.2$ ./binary_search
Element 8 found at index 3.
sh-5.2$ □
```

The above is the image of the program and output of Program 2.14.

```

C 15_file_io.c > ...
1  #include <stdio.h>
2  #include <stdlib.h>
3
4  int main(void) {
5      FILE *fp = fopen("output.txt", "w");
6      if (fp == NULL) {
7          printf("Error opening file for writing.\n");
8          return 1;
9      }
10     fprintf(fp, "Hello, file!\n");
11     fclose(fp);
12
13     fp = fopen("output.txt", "r");
14     if (fp == NULL) {
15         printf("Error opening file for reading.\n");
16         return 1;
17     }
18
19     char buffer[100];
20     while (fgets(buffer, sizeof(buffer), fp) != NULL) {
21         printf("%s", buffer);
22     }
23     fclose(fp);
24
25     return 0;
26 }
27

```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```

sh-5.2$ gcc 15_file_io.c -o file_io
sh-5.2$ ./file_io
Hello, file!
sh-5.2$ █

```

The above is the image of the program and output of Program 2.15.