

DATA STRUCTURES

NAME : B.INDRA NEELAKNATA

REG NO : 192365047

COURSE CODE : CSA0312

TOPIC : ARRAY & LINKED LIST

#smallest positive number missing from the array.

INPUT :-

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

```
int segregate(int arr[], int size) {
```

```
    int j = 0, i;
```

```
    for (i = 0; i < size; i++) {
```

```
        if (arr[i] <= 0) {
```

```
            int temp = arr[i];
```

```
            arr[i] = arr[j];
```

```
            arr[j] = temp;
```

```
            j++;
```

```
        }
```

```
    }
```

```
    return j;  
}
```

```
int findMissingPositive(int arr[], int size) {  
    int i;  
    for (i = 0; i < size; i++) {  
        int val = abs(arr[i]);  
        if (val- 1 < size && arr[val- 1] > 0) {  
            arr[val- 1] = -arr[val- 1];  
        }  
    }  
    for (i = 0; i < size; i++) {  
        if (arr[i] > 0)  
            return i + 1;  
    }  
    return size + 1;  
}
```

```
int findMissing(int arr[], int size) {  
    int shift = segregate(arr, size);  
    return findMissingPositive(arr + shift, size- shift);  
}
```

```
int main() {  
    int arr[] = {2, 3, 7, 6, 8, -1, -10, 15};  
    int size = sizeof(arr) / sizeof(arr[0]);
```

```
int missing = findMissing(arr, size);  
printf("The smallest positive missing number is %d\n", missing);  
return 0;  
}
```

OUTPUT :-

```
The smallest positive missing number is 1  
  
=== Code Execution Successful ===
```

#Finding missing element

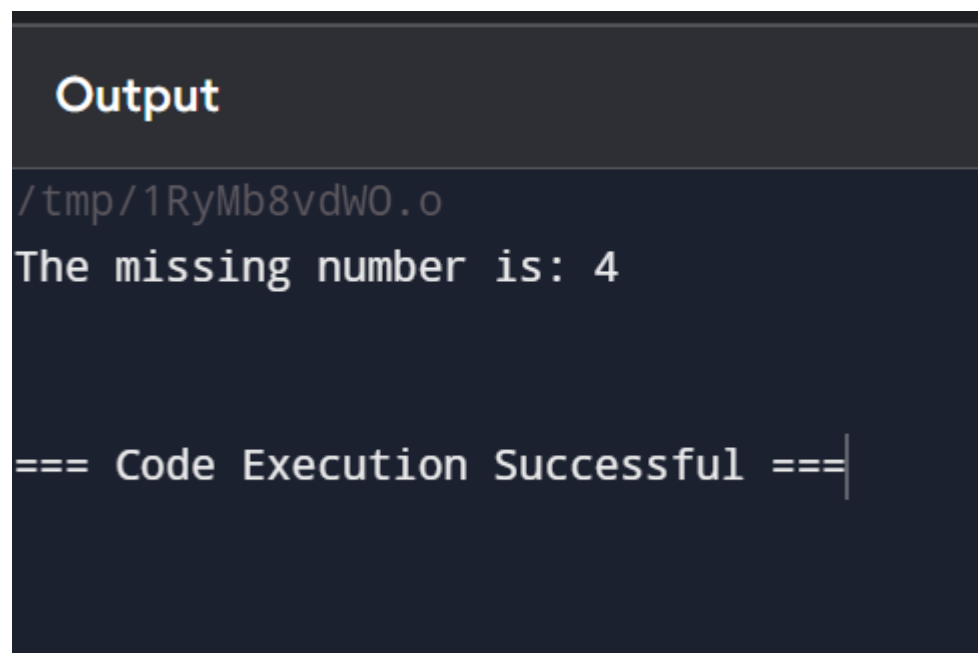
INPUT :-

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

```
int findMissing(int arr[], int N) {  
    int totalSum = (N * (N + 1)) / 2;  
    int arraySum = 0;  
  
    for (int i = 0; i < N- 1; i++) {  
        arraySum += arr[i];  
    }  
  
    return totalSum- arraySum;  
}
```

```
int main() {  
    int N = 5;  
    int arr[] = {1, 2, 3, 5};  
  
    int missing = findMissing(arr, N);  
    printf("The missing number is: %d\n", missing);  
  
    return 0;  
}
```

OUTPUT :-



```
Output  
/tmp/1RyMb8vdwO.o  
The missing number is: 4  
  
=== Code Execution Successful ===
```

#Finding odd number in linked list

INPUT :-

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

```
struct Node {  
    int data;  
    struct Node* next;  
};
```

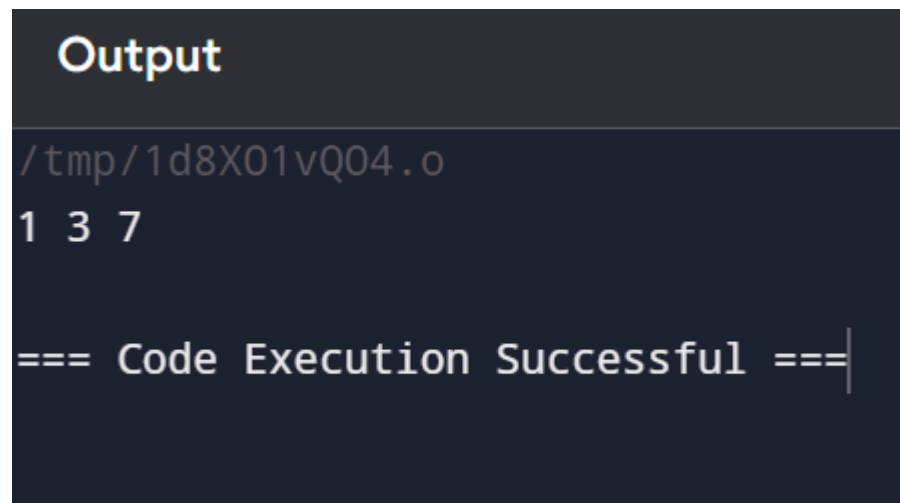
```
struct Node* newNode(int data) {  
    struct Node* node = (struct Node*)malloc(sizeof(struct Node));  
    node->data = data;  
    node->next = NULL;  
    return node;  
}
```

```
void printOddNumbers(struct Node* head) {  
    struct Node* temp = head;  
    while (temp != NULL) {  
        if (temp->data % 2 != 0) {  
            printf("%d ", temp->data);  
        }  
        temp = temp->next;  
    }  
}
```

```
int main() {  
    struct Node* head = newNode(1);  
    head->next = newNode(2);
```

```
head->next->next = newNode(3);  
head->next->next->next = newNode(7);  
  
printOddNumbers(head);  
  
return 0;  
}
```

OUTPUT :-

A screenshot of a terminal window with a dark background. The title bar at the top says "Output" in white. Below the title bar, the file path "/tmp/1d8X01vQ04.o" is shown in a light gray font. The output of the program is "1 3 7" in white. At the bottom, a status message "=== Code Execution Successful ===" is displayed in white, followed by a vertical cursor line.

```
Output  
/tmp/1d8X01vQ04.o  
1 3 7  
=== Code Execution Successful ===
```