

6. Implement a C Program Given an array of reg nos need to search for particular reg no

Code

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

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

int main() {
    // Create first list: 2 -> 3 -> 4
    struct Node *a1 = malloc(sizeof(struct Node));
    struct Node *a2 = malloc(sizeof(struct Node));
    struct Node *a3 = malloc(sizeof(struct Node));
    a1->data = 2; a1->next = a2;
    a2->data = 3; a2->next = a3;
    a3->data = 4; a3->next = NULL;

    // Create second list: 5 -> 6 -> 7
    struct Node *b1 = malloc(sizeof(struct Node));
    struct Node *b2 = malloc(sizeof(struct Node));
    struct Node *b3 = malloc(sizeof(struct Node));
    b1->data = 5; b1->next = b2;
    b2->data = 6; b2->next = b3;
    b3->data = 7; b3->next = NULL;

    // Merge: attach end of first to start of second
}
```

```

struct Node *temp = a1;

while (temp->next != NULL)

    temp = temp->next;

    temp->next = b1;

// Print merged list

printf("Merged list: ");

temp = a1;

while (temp != NULL) {

    printf("%d ", temp->data);

    temp = temp->next;

}

printf("\n");

return 0;
}

```

Output

```

main.c | [main.c] | Code | Run | Output
1 //main.c
2 #include <stdio.h>
3
4 struct Node {
5     int data;
6     struct Node *next;
7 };
8
9 int main() {
10     // Create first list: 2 -> 3 -> 4
11     struct Node *a1 = malloc(sizeof(struct Node));
12     struct Node *a2 = malloc(sizeof(struct Node));
13     struct Node *a3 = malloc(sizeof(struct Node));
14     a1->data = 2; a1->next = a2;
15     a2->data = 3; a2->next = a3;
16     a3->data = 4; a3->next = NULL;
17
18     // Create second list: 5 -> 6 -> 7
19     struct Node *b1 = malloc(sizeof(struct Node));
20     struct Node *b2 = malloc(sizeof(struct Node));
21     struct Node *b3 = malloc(sizeof(struct Node));
22     b1->data = 5; b1->next = b2;
23     b2->data = 6; b2->next = b3;
24     b3->data = 7; b3->next = NULL;
25
26     // Merge: attach end of first to start of second
27     struct Node *temp = a1;
28     while (temp->next != NULL)
29         temp = temp->next;
30     temp->next = b1;
31
32     // Print merged list
33     printf("Merged list: ");
34     temp = a1;
35     while (temp != NULL) {
36         printf("%d ", temp->data);
37         temp = temp->next;
38     }
39     printf("\n");
40
41     return 0;
42 }

```

Output

```

Merged list: 2 3 4 5 6 7
*** Code Execution Successful ***

```