1- Write a program to create an empty linked list and add three nodes with data values 2, 4, and 6. Print the elements of the linked list.

اكتب برنامجًا لإنشاء linked list فارغة وإضافة ثلاث nodes بقيم البيانات 2 و4 و6. اطبع عناصر linked list.

Output

```
2 4 6
```

Solution

```c
// www.gammal.tech

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

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

int main() {
    struct node *head, *temp;
    head = (struct node*)malloc(sizeof(struct node));
    head->data = 2;
    head->next = (struct node*)malloc(sizeof(struct node));
    head->next->data = 4;
    head->next->next = (struct node*)malloc(sizeof(struct node));
    head->next->next->data = 6;
    head->next->next->next = NULL;

    temp = head;
    while (temp != NULL) {
        printf("%d ", temp->data);
        temp = temp->next;
    }

    return 0;
}
```

2- Write a program to add a node with data value 8 at the end of the linked list created in the previous program. Print the updated linked list.

اكتب برنامجًا لإضافة node بقيمة البيانات 8 في نهاية القائمة المرتبطة التي تم إنشاؤها في البرنامج السابق. طباعة linked list المحدثة.

Output

```
2 4 6 8
```

Solution

```c
// www.gammal.tech

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

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

int main() {
    struct node *head, *temp;
    head = (struct node*)malloc(sizeof(struct node));
    head->data = 2;
    head->next = (struct node*)malloc(sizeof(struct node));
    head->next->data = 4;
    head->next->next = (struct node*)malloc(sizeof(struct node));
    head->next->next->data = 6;
    head->next->next->next = NULL;

    // Adding a new node with data value 8 at the end
    temp = head;
    while (temp->next != NULL) {
        temp = temp->next;
    }
    temp->next = (struct node*)malloc(sizeof(struct node));
    temp->next->data = 8;
    temp->next->next = NULL;

    // Print the updated linked list
    temp = head;
    while (temp != NULL) {
        printf("%d ", temp->data);
        temp = temp->next;
    }

    return 0;
}
```

3- Write a program to find and print the length (number of nodes) of the linked list.

اكتب برنامجًا للعثور على طول (عدد node) لlinked list وطباعتها.

Output

```
Length of the linked list: 3
```

Solution

```c
// www.gammal.tech

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

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

int main() {
    struct node *head, *temp;
    head = (struct node*)malloc(sizeof(struct node));
    head->data = 2;
    head->next = (struct node*)malloc(sizeof(struct node));
    head->next->data = 3;
    head->next->next = (struct node*)malloc(sizeof(struct node));
    head->next->next->data = 4;
    head->next->next->next = NULL;

    // Find the length of the linked list
    int length = 0;
    temp = head;
    while (temp != NULL) {
        length++;
        temp = temp->next;
    }

    // Print the length
    printf("Length of the linked list: %d\n", length);

    // Free allocated memory
    temp = head;
    while (temp != NULL) {
        struct node* nextNode = temp->next;
        free(temp);
        temp = nextNode;
    }

    return 0;
}
```

4- Write a program to check if the linked list contains a node with data value 5. Print "Found" if it exists, otherwise print "Not Found".

اكتب برنامجًا للتحقق مما إذا كانت linked list تحتوي على nodeبقيمة البيانات 5. اطبع "تم العثور عليه" إذا كان موجودًا، وإلا فاطبع "لم يتم العثور عليه".

Output

```
Not Found
```

Solution

```c
// www.gammal.tech

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

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

int main() {
    struct node *head, *temp;
    head = (struct node*)malloc(sizeof(struct node));
    head->data = 2;
    head->next = (struct node*)malloc(sizeof(struct node));
    head->next->data = 3;
    head->next->next = (struct node*)malloc(sizeof(struct node));
    head->next->next->data = 4;
    head->next->next->next = NULL;

    // Check if the linked list contains a node with data value 5
    int found = 0;
    temp = head;
    while (temp != NULL) {
        if (temp->data == 5) {
            found = 1;
            break;
        }
        temp = temp->next;
    }

    // Print the result
    if (found) {
        printf("Found\n");
    } else {
        printf("Not Found\n");
    }

    // Free allocated memory
    temp = head;
    while (temp != NULL) {
        struct node* nextNode = temp->next;
        free(temp);
        temp = nextNode;
    }

    return 0;
}
```

5- Write a program to reverse the linked list. Print the reversed linked list.

اكتب برنامجًا لعكس القائمة المرتبطة. طباعة القائمة المرتبطة المعكوسة.

Output

```
6 4 2
```

Solution

```c
// www.gammal.tech

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

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

int main() {
    struct node *head, *temp, *prev, *next;

    // Create a linked list with three nodes
    head = (struct node*)malloc(sizeof(struct node));
    head->data = 2;
    head->next = (struct node*)malloc(sizeof(struct node));
    head->next->data = 4;
    head->next->next = (struct node*)malloc(sizeof(struct node));
    head->next->next->data = 6;
    head->next->next->next = NULL;

    // Reverse the linked list
    prev = NULL;
    temp = head;
    while (temp != NULL) {
        next = temp->next;
        temp->next = prev;
        prev = temp;
        temp = next;
    }
    head = prev;

    // Print the reversed linked list
    temp = head;
    while (temp != NULL) {
        printf("%d ", temp->data);
        temp = temp->next;
    }

    // Free allocated memory
    temp = head;
    while (temp != NULL) {
        struct node* nextNode = temp->next;
        free(temp);
        temp = nextNode;
    }

    return 0;
}
```

6- Write a program to delete all nodes from the linked list. Print the empty linked list.

اكتب برنامجًا لحذف كافة node من linked list. طباعة linked list الفارغة.

Output

```
Empty linked list
```

Solution

```c
// www.gammal.tech

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

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

int main() {
    struct node *head, *temp;

    // Create a linked list with three nodes
    head = (struct node*)malloc(sizeof(struct node));
    head->data = 2;
    head->next = (struct node*)malloc(sizeof(struct node));
    head->next->data = 4;
    head->next->next = (struct node*)malloc(sizeof(struct node));
    head->next->next->data = 6;
    head->next->next->next = NULL;

    // Delete all nodes from the linked list
    temp = head;
    while (temp != NULL) {
        struct node* nextNode = temp->next;
        free(temp);
        temp = nextNode;
    }

    // Print the empty linked list
    printf("Empty linked list\n");

    return 0;
}
```

7- Write a program to find and print the sum of all data values in the linked list.

اكتب برنامجًا للعثور على مجموع كل قيم البيانات في linked list وطباعته.

Output

```
Sum of data values: 12
```

Solution

```c
// www.gammal.tech

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

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

int main() {
    struct node *head, *temp;

    // Create a linked list with three nodes
    head = (struct node*)malloc(sizeof(struct node));
    head->data = 2;
    head->next = (struct node*)malloc(sizeof(struct node));
    head->next->data = 4;
    head->next->next = (struct node*)malloc(sizeof(struct node));
    head->next->next->data = 6;
    head->next->next->next = NULL;

    // Find the sum of all data values in the linked list
    int sum = 0;
    temp = head;
    while (temp != NULL) {
        sum += temp->data;
        temp = temp->next;
    }

    // Print the sum
    printf("Sum of data values: %d\n", sum);

    // Free allocated memory
    temp = head;
    while (temp != NULL) {
        struct node* nextNode = temp->next;
        free(temp);
        temp = nextNode;
    }

    return 0;
}
```

8- Write a program to check if the linked list is empty. Print "Empty" if it is, otherwise print "Not Empty".

اكتب برنامجًا للتحقق مما إذا كانت linked list فارغة. اطبع "فارغًا" إذا كان كذلك، وإلا فاطبع "ليس فارغًا".

Output

```
Empty
```

Solution

```c
// www.gammal.tech

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

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

int main() {
    struct node *head;

    // Create an empty linked list
    head = NULL;

    // Check if the linked list is empty
    if (head == NULL) {
        printf("Empty\n");
    } else {
        printf("Not Empty\n");
    }

    // Free allocated memory (none in this case)

    return 0;
}
```

9- Write a program to find and print the minimum data value in the linked list.

اكتب برنامجًا للعثور على الحد الأدنى من قيمة البيانات في linked list وطباعتها.

Output

```
Minimum data value: 2
```

Solution

```c
// www.gammal.tech

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

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

int main() {
    struct node *head, *temp;

    // Create a linked list with three nodes
    head = (struct node*)malloc(sizeof(struct node));
    head->data = 2;
    head->next = (struct node*)malloc(sizeof(struct node));
    head->next->data = 4;
    head->next->next = (struct node*)malloc(sizeof(struct node));
    head->next->next->data = 6;
    head->next->next->next = NULL;

    // Find the minimum data value in the linked list
    int min = INT_MAX;
    temp = head;
    while (temp != NULL) {
        if (temp->data < min) {
            min = temp->data;
        }
        temp = temp->next;
    }

    // Print the minimum data value
    if (min == INT_MAX) {
        printf("Linked list is empty.\n");
    } else {
        printf("Minimum data value: %d\n", min);
    }

    // Free allocated memory
    temp = head;
    while (temp != NULL) {
        struct node* nextNode = temp->next;
        free(temp);
        temp = nextNode;
    }

    return 0;
}
```

10- Write a program to print the data values of the linked list until a node with data value 5 is encountered.

اكتب برنامجًا لطباعة قيم بيانات linked list حتى يتم العثور على node بقيمة البيانات 5.

Output

```
2 4 6
```

Solution

```c
// www.gammal.tech

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

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

int main() {
    struct node *head, *temp;

    // Create a linked list with three nodes
    head = (struct node*)malloc(sizeof(struct node));
    head->data = 2;
    head->next = (struct node*)malloc(sizeof(struct node));
    head->next->data = 4;
    head->next->next = (struct node*)malloc(sizeof(struct node));
    head->next->next->data = 6;
    head->next->next->next = NULL;

    // Print the data values until a node with data value 5 is encountered
    temp = head;
    while (temp != NULL) {
        printf("%d ", temp->data);
        if (temp->data == 5) {
            break;
        }
        temp = temp->next;
    }

    // Free allocated memory
    temp = head;
    while (temp != NULL) {
        struct node* nextNode = temp->next;
        free(temp);
        temp = nextNode;
    }

    return 0;
}
```