2110213561 Manar Mohamed

----- DATA STRUCTURE H.W-----

1 -
```
#include <stdio.h>
void addNumbers() {
    int numbers[100];
    int num, index = 0;
    while (1) {
        printf("Enter a number (-1 to stop): ");
        scanf("%d", &num);
        if (num == -1) {
            break;
        }
        if (num % 2 != 0) {
            for (int i = index; i >= 0; i--) {
                numbers[i + 1] = numbers[i];
            }
            numbers[0] = num;
            index++;
        } else {
            numbers[index] = num;
            index++;
        }
    }
    printf("Modified list: ");
    for (int i = 0; i < index; i++) {
        printf("%d ", numbers[i]);
    }
}

int main() {
    addNumbers();
    return 0;
}
2-   #include <stdio.h>
     #include <stdlib.h>
void bubbleSort(int arr[], int n) {
    int temp;
    for (int i = 0; i < n - 1; i++) {
        for (int j = 0; j < n - i - 1; j++) {
            if (arr[j] < arr[j + 1]) {
                // Swap arr[j] and arr[j + 1]
                temp = arr[j];
                arr[j] = arr[j + 1];
                arr[j + 1] = temp;
            }
        }
    }
}

int main() {
    int numbers[200]; numbers
    int num, index = 0;
     srand(time(0));
    for (int i = 0; i < 100; i++) {
        numbers[index++] = rand() % 1000; // Generate random numbers between 0 and 999
    }

       while (1) {
        printf("Enter a number (-1 to stop): ");
        scanf("%d", &num);

        if (num == -1) {
            break;
        }
        numbers[index++] = num;
    }
     bubbleSort(numbers, index);
     printf("Sorted numbers in descending order:\n");
    for (int i = 0; i < index; i++) {
        printf("%d ", numbers[i]);
    }

    return 0;
}
```

```
3- #include <stdio.h>
int main() {
    int start = 54;
```

```
   int end = 102;
   int step = 4;
   printf("%d", start);
   for (int i = start + step; i <= end; i += step) {
       printf("->%d", i);
   }
   printf("\n");
   return 0;
}
```

```
4- #include <stdio.h>
#include <stdlib.h>
#include <string.h>
 struct Student {
   int number;
   char name[50];
   int age;
   int studentID;
   struct Student* next;
};
 struct Student* createStudent(int number, const char* name, int age, int studentID) {
   struct Student* newStudent = (struct Student*)malloc(sizeof(struct Student));
   if (newStudent != NULL) {
       newStudent->number = number;
       strncpy(newStudent->name, name, sizeof(newStudent->name));
       newStudent->age = age;
       newStudent->studentID = studentID;
```

```
        newStudent->next = NULL;
    }
    return newStudent;
}

void insertStudent(struct Student** head, int number, const char* name, int age, int studentID)
{
    struct Student* newStudent = createStudent(number, name, age, studentID);
    if (newStudent != NULL) {
        newStudent->next = *head;
        *head = newStudent;
    }
}

void displayStudents(struct Student* head) {
    struct Student* current = head;
    int count = 0;
    printf("Student Information:\n");
    while (current != NULL) {
        printf("%d- %s %d %d\n", current->number, current->name, current->age, current-
>studentID);
        current = current->next;
        count++;
    }
    printf("Total number of students: %d\n", count);
}
 void freeStudents(struct Student* head) {
    struct Student* current = head;
    struct Student* next;

    while (current != NULL) {
        next = current->next;
        free(current);
   current = next;
    }
}
int main() {
    struct Student* head = NULL;
     insertStudent(&head, 1, "Saliha", 27, 201);
    insertStudent(&head, 2, "Ece", 19, 203);
     displayStudents(head);
     freeStudents(head);
    return 0;
}
```

5- #include <stdio.h>

```
#include <stdlib.h>
#include <string.h>
 struct Student {
    int number;
    char name[50];
    int age;
    int studentID;
    struct Student* next;
};
 struct Student* createStudent(int number, const char* name, int age, int studentID) {
    struct Student* newStudent = (struct Student*)malloc(sizeof(struct Student));
    if (newStudent != NULL) {
        newStudent->number = number;
        strncpy(newStudent->name, name, sizeof(newStudent->name));
        newStudent->age = age;
        newStudent->studentID = studentID;
        newStudent->next = NULL;
    }
    return newStudent;
}
 void insertStudent(struct Student** head, int number, const char* name, int age, int
studentID) {
    struct Student* newStudent = createStudent(number, name, age, studentID);
    if (newStudent != NULL) {
        newStudent->next = *head;
        *head = newStudent;
    }
}
 struct Student* searchByName(struct Student* head, const char* name) {
    struct Student* current = head;
    while (current != NULL) {
        if (strcmp(current->name, name) == 0) {
            return current; // Found a match
        }
        current = current->next;
    }
    return NULL; // Name not found
```

```
}
 void freeStudents(struct Student* head) {
    struct Student* current = head;
    struct Student* next;

    while (current != NULL) {
        next = current->next;
        free(current);
        current = next;
    }
}

int main() {
    struct Student* head = NULL;
     insertStudent(&head, 1, "Saliha", 27, 201);
    insertStudent(&head, 2, "Ece", 19, 203);
     const char* searchName = "Saliha";
    struct Student* result = searchByName(head, searchName);

    if (result != NULL) {
        printf("Student found - Number: %d, Name: %s, Age: %d, ID: %d\n", result->number, 
result->name, result->age, result->studentID);
    } else {
        printf("Student with name '%s' not found.\n", searchName);
    }
     freeStudents(head);
    return 0;
}
```

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

#include <stdlib.h>

#include <string.h>

 struct Student {

    int number;

    char name[50];

    int age;

    int studentID;

    struct Student* next;

};

 struct Student* createStudent(int number, const char* name, int age, int studentID) {

    struct Student* newStudent = (struct Student*)malloc(sizeof(struct Student));

    if (newStudent != NULL) {

        newStudent->number = number;

        strncpy(newStudent->name, name, sizeof(newStudent->name));

        newStudent->age = age;

        newStudent->studentID = studentID;

        newStudent->next = NULL;

    }

    return newStudent;

}

 void insertStudent(struct Student** head, int number, const char* name, int age, int
studentID) {

    struct Student* newStudent = createStudent(number, name, age, studentID);

    if (newStudent != NULL) {

        newStudent->next = *head;

        *head = newStudent;
    }
}

struct Student* searchByName(struct Student* head, const char* name) {
    struct Student* current = head;
    struct Student* prev = NULL;

    while (current != NULL) {
        if (strcmp(current->name, name) == 0) {
            return prev; // Return the node before the match
        }
        prev = current;
        current = current->next;
    }
    return NULL; // Name not found
}
 void deleteNextNode(struct Student* nodeBefore) {
    if (nodeBefore != NULL && nodeBefore->next != NULL) {
        struct Student* temp = nodeBefore->next;
        nodeBefore->next = temp->next;
        free(temp);
    }
}
 void freeStudents(struct Student* head) {
    struct Student* current = head;
    struct Student* next;
    while (current != NULL) {
        next = current->next;

        free(current);

        current = next;

    }

}


int main() {

    struct Student* head = NULL;

     insertStudent(&head, 1, "Saliha", 27, 201);

    insertStudent(&head, 2, "Ece", 19, 203);

    insertStudent(&head, 3, "Ali", 22, 205);

     const char* searchName = "Saliha";

    struct Student* nodeBefore = searchByName(head, searchName);


    if (nodeBefore != NULL) {

        printf("Deleting the node after %s\n", nodeBefore->name);

        deleteNextNode(nodeBefore);

    } else {

        printf("Student with name '%s' not found.\n", searchName);

    }


    struct Student* current = head;

    while (current != NULL) {

        printf("%d- %s %d %d\n", current->number, current->name, current->age, current-
>studentID);

        current = current->next;

    }

    freeStudents(head);
    return 0;

}

7- #include <stdio.h>

#include <stdlib.h>

#include <string.h>


struct Student {

    int number;

    char name[50];

    int age;

    int studentID;

    struct Student* next;

};

 struct Student* createStudent(int number, const char* name, int age, int studentID) {

    struct Student* newStudent = (struct Student*)malloc(sizeof(struct Student));

    if (newStudent != NULL) {

        newStudent->number = number;

        strncpy(newStudent->name, name, sizeof(newStudent->name));

        newStudent->age = age;

        newStudent->studentID = studentID;

        newStudent->next = NULL;

    }

    return newStudent;

}

 void insertStudent(struct Student** head, int number, const char* name, int age, int
studentID) {

    struct Student* newStudent = createStudent(number, name, age, studentID);
    if (newStudent != NULL) {
        newStudent->next = *head;
        *head = newStudent;
    }
}
 void printLongestName(struct Student* head) {
    if (head == NULL) {
        printf("The list is empty.\n");
        return;
    }

    struct Student* current = head;
    char longestName[50] = "";
    while (current != NULL) {
        if (strlen(current->name) > strlen(longestName)) {
            strncpy(longestName, current->name, sizeof(longestName));
        }
        current = current->next;
    }

    printf("The longest name in the list: %s\n", longestName);
    printf("Length: %d\n", (int)strlen(longestName));
}
 void freeStudents(struct Student* head) {
    struct Student* current = head;
    struct Student* next;
    while (current != NULL) {
        next = current->next;
        free(current);
        current = next;
    }
}

int main() {
    struct Student* head = NULL;
     insertStudent(&head, 1, "Alice", 22, 101);
    insertStudent(&head, 2, "Bob", 25, 102);
    insertStudent(&head, 3, "Abdurrahmangazi", 30, 103);
    insertStudent(&head, 4, "Charlie", 28, 104);
     printLongestName(head);
    freeStudents(head);

    return 0;
}
```