**Міністерство освіти і науки України  
Національний технічний університет України  
«Київський політехнічний інститут імені Ігоря Сікорського»  
Факультет інформатики та обчислювальної техніки  
Кафедра обчислювальної техніки**

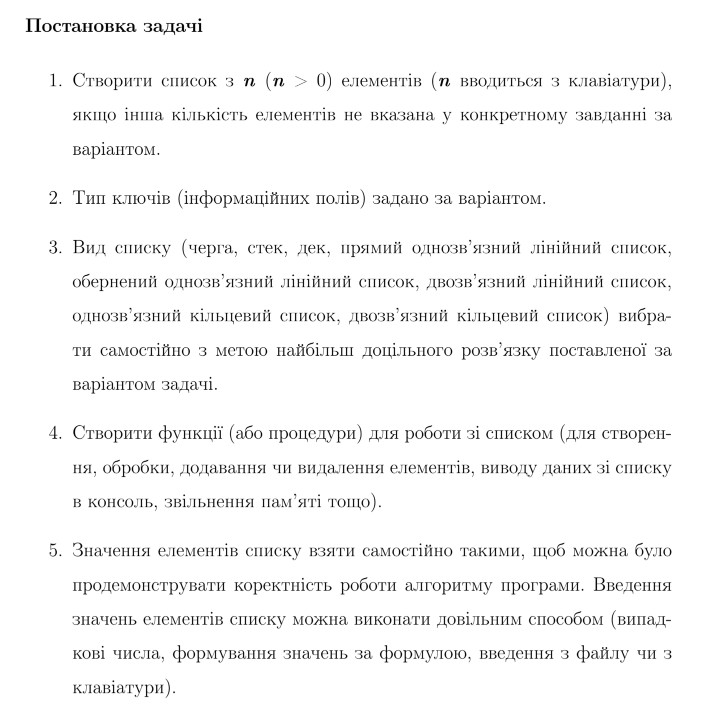
**Лабораторна робота №2**

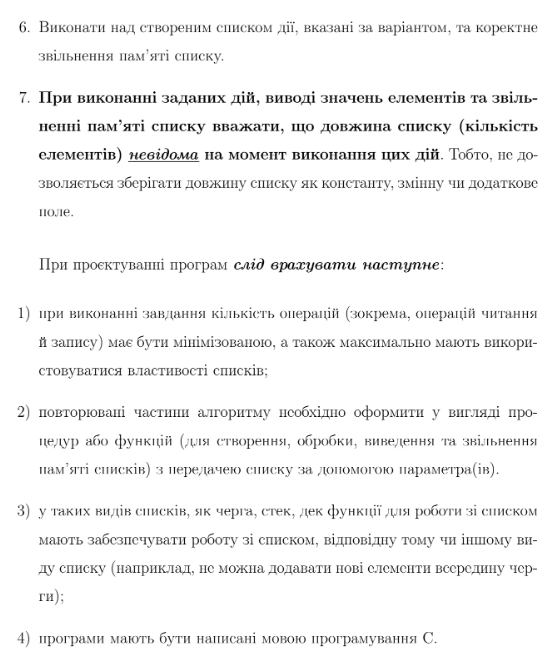
з дисципліни  
«Алгоритми і структури даних»

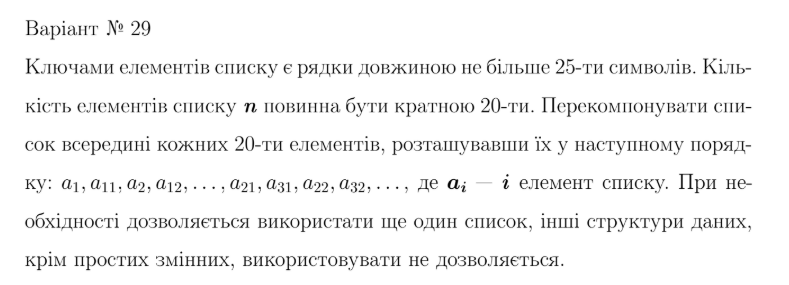
Виконав: Перевірив:

студент групи ІМ-42 Сергієнко А. М.  
Федоренко Іван Русланович  
номер у списку групи: 29

Київ 2025







**Код програми:**

#include <stdio.h>

#include <stdlib.h>

#include <string.h>

typedef struct Node {

    char data[26];

    struct Node\* next;

} Node;

Node\* create\_node(const char\* str) {

    Node\* new\_node = (Node\*)malloc(sizeof(Node));

    if (new\_node) {

        strncpy(new\_node->data, str, 25);

        new\_node->data[25] = '\0';

        new\_node->next = NULL;

    }

    return new\_node;

}

void append(Node\*\* head, const char\* str) {

    Node\* new\_node = create\_node(str);

    if (\*head == NULL) {

        \*head = new\_node;

    } else {

        Node\* temp = \*head;

        while (temp->next) {

            temp = temp->next;

        }

        temp->next = new\_node;

    }

}

void print\_list(Node\* head) {

    Node\* temp = head;

    while (temp != NULL) {

        printf("%s ", temp->data);

        temp = temp->next;

    }

}

void free\_list(Node\* head) {

    Node\* temp;

    while (head != NULL) {

        temp = head;

        head = head->next;

        free(temp);

    }

}

Node\* rearrange(Node\* head, int n) {

    if (head == NULL || n <= 0 || n % 20 != 0) {

        return head;

    }

    Node\* new\_head = NULL;

    Node\* new\_tail = NULL;

    Node\* current = head;

    for (int i = 0; i < n / 20; i++) {

        Node\* block\_heads[20] = {NULL};

        for (int j = 0; j < 20; j++) {

            if (current != NULL) {

                Node\* temp = current;

                current = current->next;

                temp->next = NULL;

                block\_heads[j] = temp;

            } else {

                break;

            }

        }

        for (int j = 0; j < 10; j++) {

            if (block\_heads[j] != NULL) {

                if (new\_head == NULL) {

                    new\_head = block\_heads[j];

                    new\_tail = block\_heads[j];

                } else {

                    new\_tail->next = block\_heads[j];

                    new\_tail = block\_heads[j];

                }

            }

            if (block\_heads[j + 10] != NULL) {

                if (new\_head == NULL) {

                    new\_head = block\_heads[j + 10];

                    new\_tail = block\_heads[j + 10];

                } else {

                    new\_tail->next = block\_heads[j + 10];

                    new\_tail = block\_heads[j + 10];

                }

            }

        }

    }

    return new\_head;

}

int main() {

    int n;

    printf("Enter the number of elements (multiple of 20): ");

    scanf("%d", &n);

    if (n <= 0 || n % 20 != 0) {

        printf("Invalid input. The number of elements should be a positive multiple of 20.\n");

        return 1;

    }

    Node\* list = NULL;

    char str[26];

    printf("Enter the elements (strings up to 25 characters each):\n");

    for (int i = 0; i < n; i++) {

        scanf("%s", str);

        append(&list, str);

    }

    printf("\nOriginal list:\n");

    print\_list(list);

    list = rearrange(list, n);

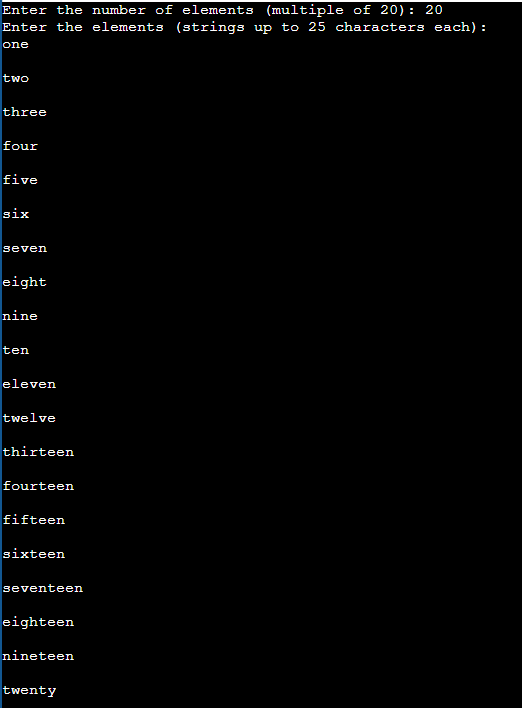
    printf("\nRearranged list:\n");

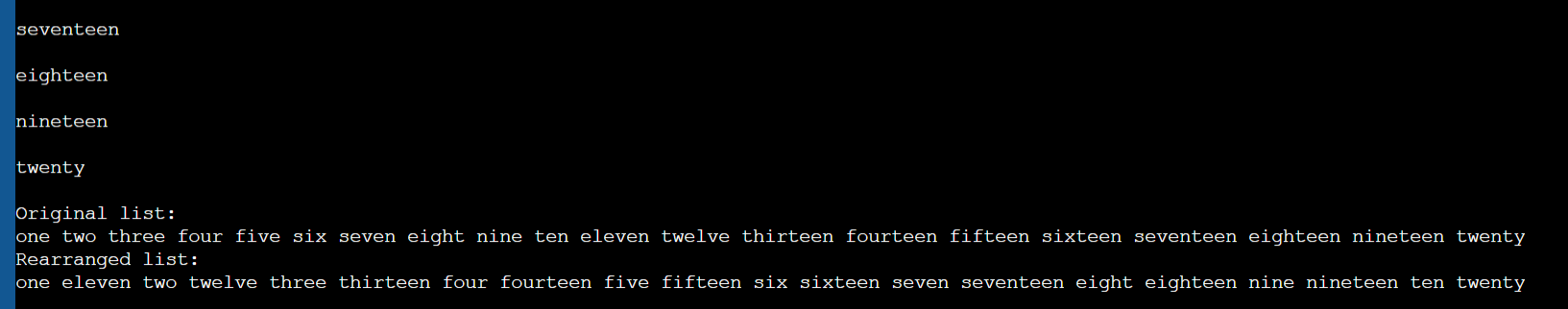
    print\_list(list);

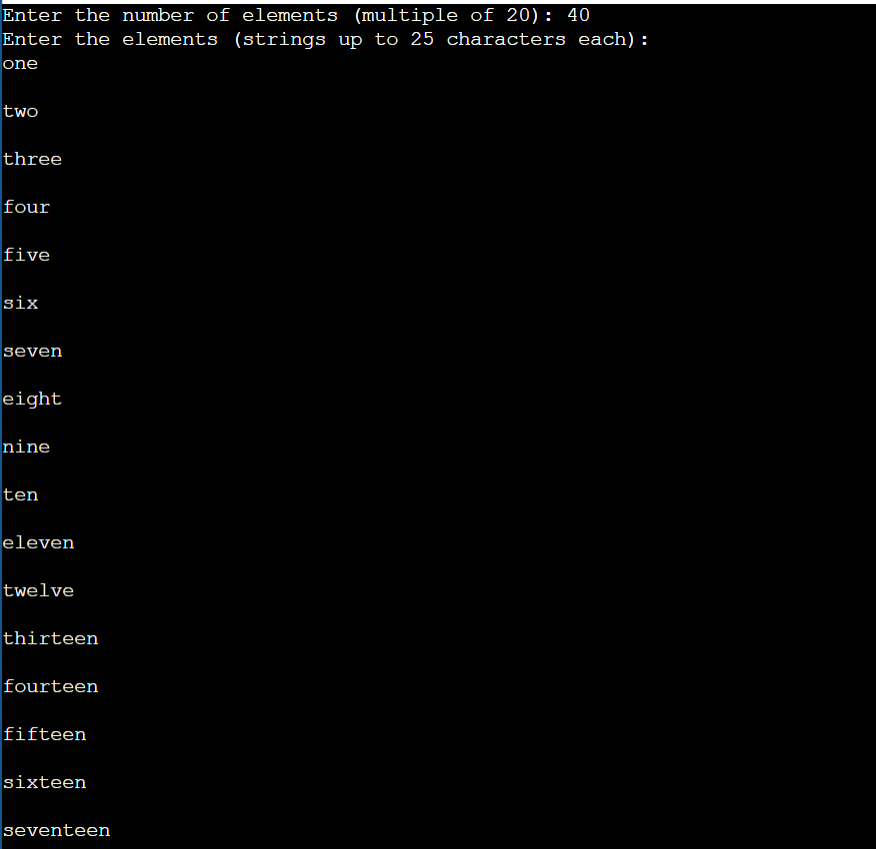
    free\_list(list);

    return 0;

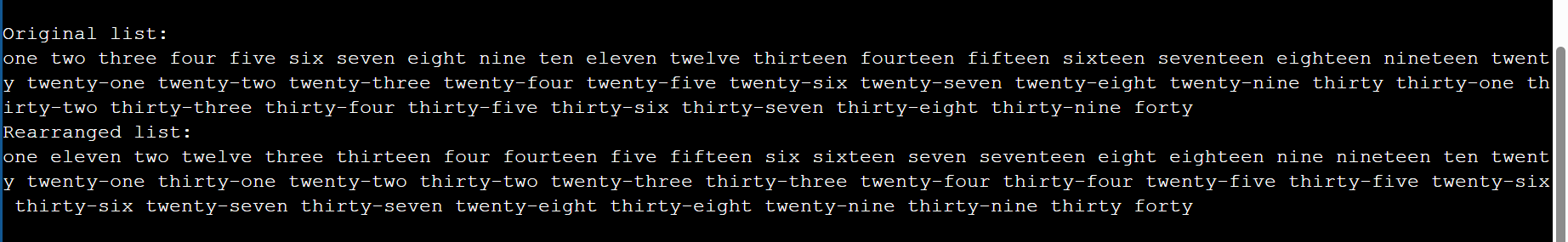
}

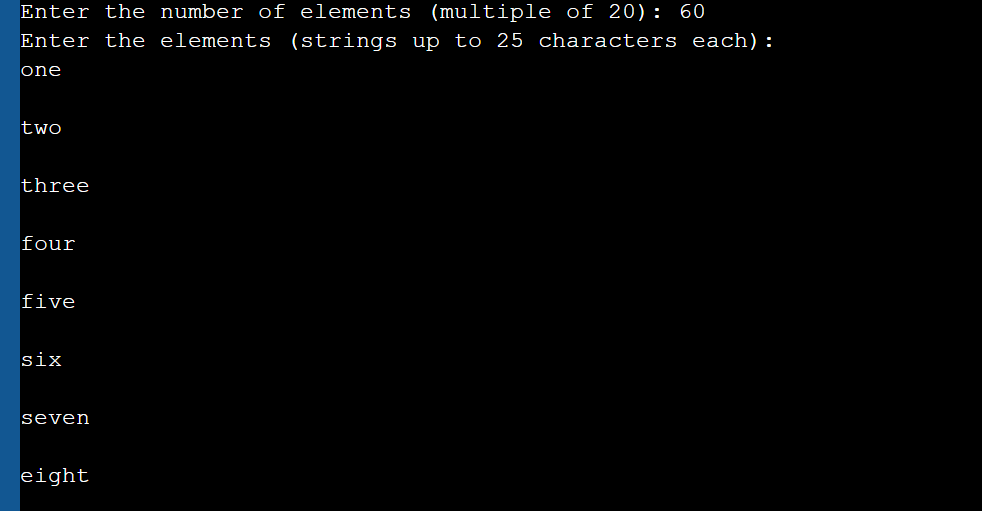
**Результати тестування:  
**



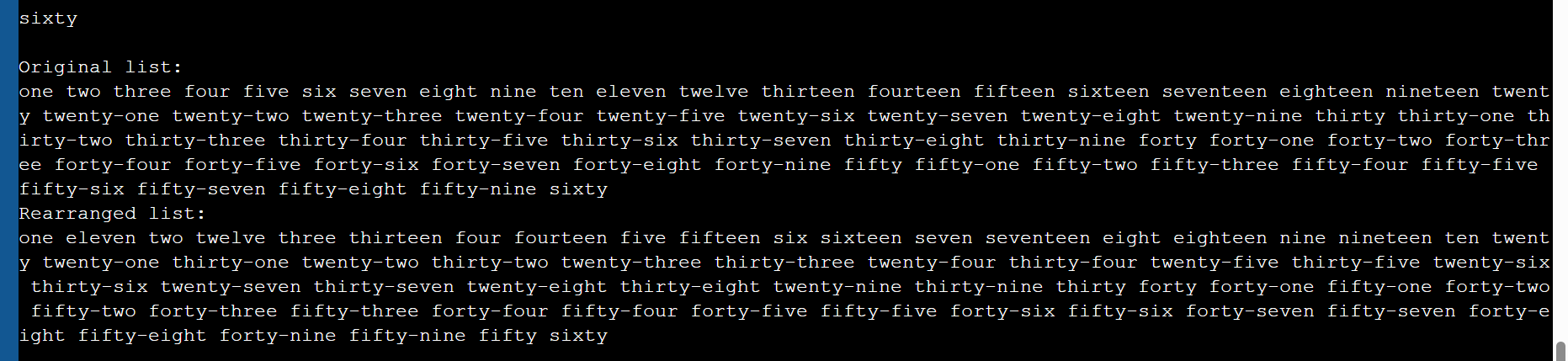


|  |  |
| --- | --- |
|  |  |





|  |  |  |
| --- | --- | --- |
|  |  |  |



**Висновки:**

Я навчився працювати з однонаправленим зв'язним списком рядків, маніпулювати списком відповідно до заданого шаблону та ефективно управляти пам'яттю. Також закріпив знання з перерозподілення списку та виведення результату, звільняючи пам'ять в кінці для уникнення витоків.