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

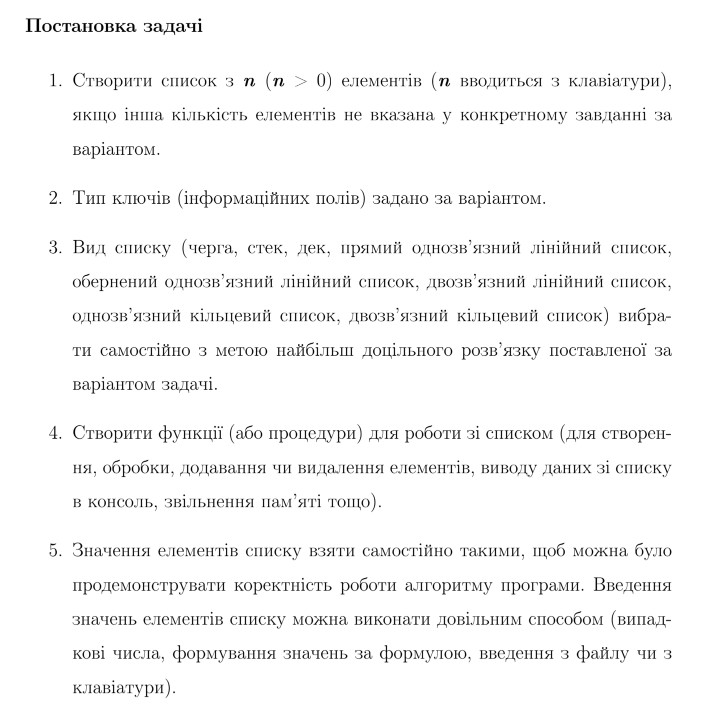
**Лабораторна робота №2**

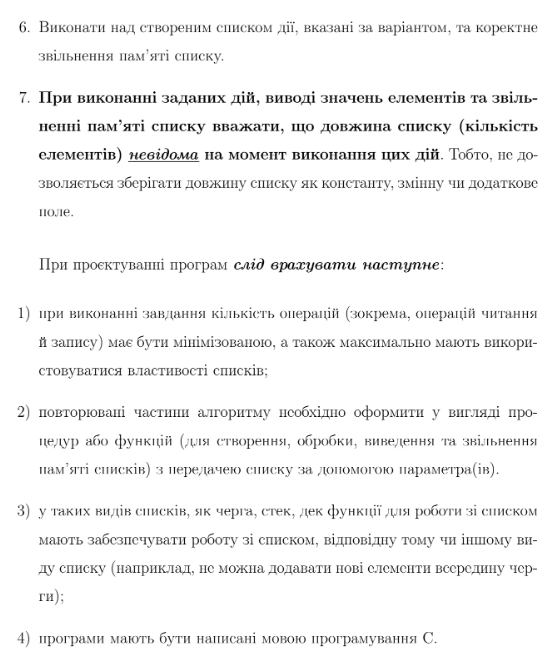
з дисципліни  
«Алгоритми і структури даних»

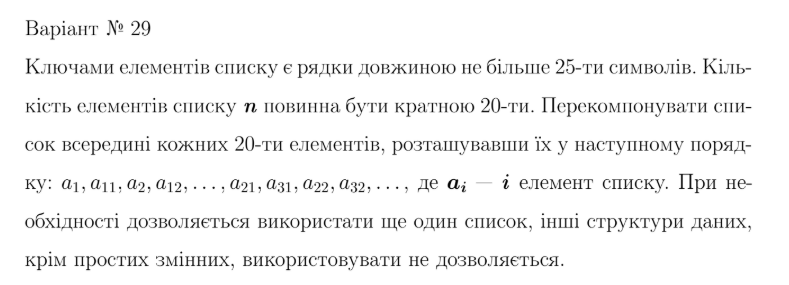
Виконав: Перевірив:

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>

#define MAX\_STRING\_LENGTH 26

#define BLOCK 20

#define HALF  (BLOCK/2)

typedef struct Node {

    char data[MAX\_STRING\_LENGTH];

    struct Node\* next;

} Node;

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

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

    if (!new\_node) {

        perror("malloc");

        exit(EXIT\_FAILURE);

    }

    strncpy(new\_node->data, str, MAX\_STRING\_LENGTH - 1);

    new\_node->data[MAX\_STRING\_LENGTH - 1] = '\0';

    new\_node->next = NULL;

    return new\_node;

}

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

    Node\* nd = create\_node(str);

    if (\*head == NULL) {

        \*head = nd;

    } else {

        Node\* t = \*head;

        while (t->next) t = t->next;

        t->next = nd;

    }

}

void print\_list(Node\* head) {

    for (; head; head = head->next) {

        printf("%s ", head->data);

    }

    printf("\n");

}

void free\_list(Node\* head) {

    while (head) {

        Node\* t = head;

        head = head->next;

        free(t);

    }

}

Node\* rearrange(Node\* head) {

    Node dummy;

    Node\* tail = &dummy;

    dummy.next = NULL;

    Node\* rest = head;

    while (rest) {

        Node\* first\_half = rest;

        Node\* mid = rest;

        for (int i = 0; i < HALF; i++) {

            if (!mid) return dummy.next;

            mid = mid->next;

        }

        Node\* block\_tail = mid;

        for (int i = 0; i < HALF - 1; i++) {

            if (!block\_tail) return dummy.next;

            block\_tail = block\_tail->next;

        }

        Node\* next\_block = block\_tail ? block\_tail->next : NULL;

        if (block\_tail) block\_tail->next = NULL;

        while (first\_half && mid) {

            Node\* first = first\_half;

            first\_half = first\_half->next;

            first->next = NULL;

            tail->next = first;

            tail = first;

            Node\* second = mid;

            mid = mid->next;

            second->next = NULL;

            tail->next = second;

            tail = second;

        }

        rest = next\_block;

    }

    return dummy.next;

}

int main(void) {

    int n;

    printf("Enter number of elements (positive multiple of %d): ", BLOCK);

    if (scanf("%d", &n) != 1 || n <= 0 || n % BLOCK != 0) {

        fprintf(stderr, "Bad input: n must be positive multiple of %d\n", BLOCK);

        return EXIT\_FAILURE;

    }

    Node\* list = NULL;

    char buf[MAX\_STRING\_LENGTH];

    printf("Enter %d strings (up to %d chars each):\n", n, MAX\_STRING\_LENGTH - 1);

    for (int i = 0; i < n; i++) {

        if (scanf("%25s", buf) != 1) {

            fprintf(stderr, "Failed to read string\n");

            free\_list(list);

            return EXIT\_FAILURE;

        }

        append(&list, buf);

    }

    printf("\nOriginal list:\n");

    print\_list(list);

    Node\* new\_list = rearrange(list);

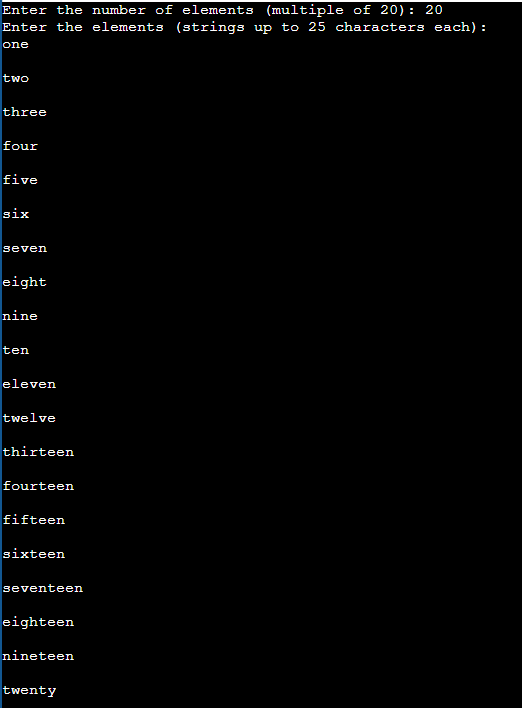
    printf("\nRearranged list:\n");

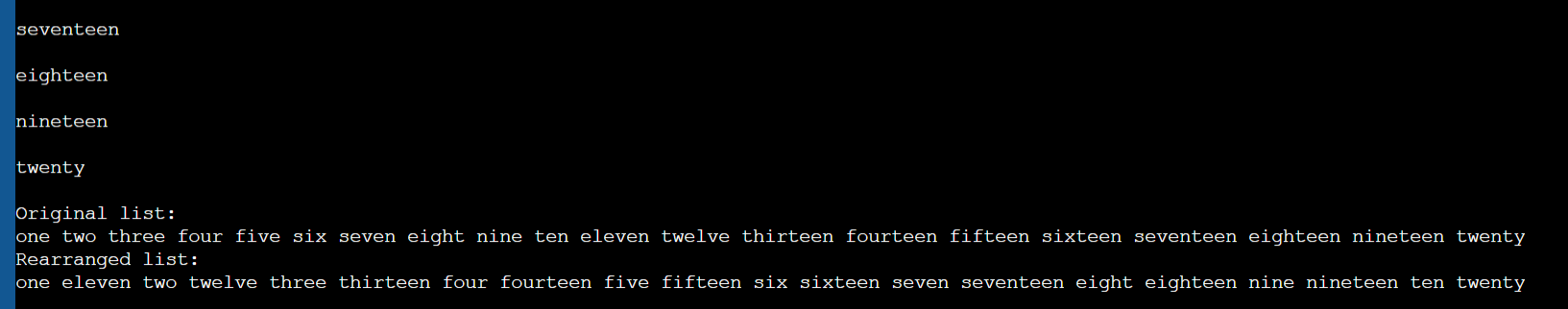
    print\_list(new\_list);

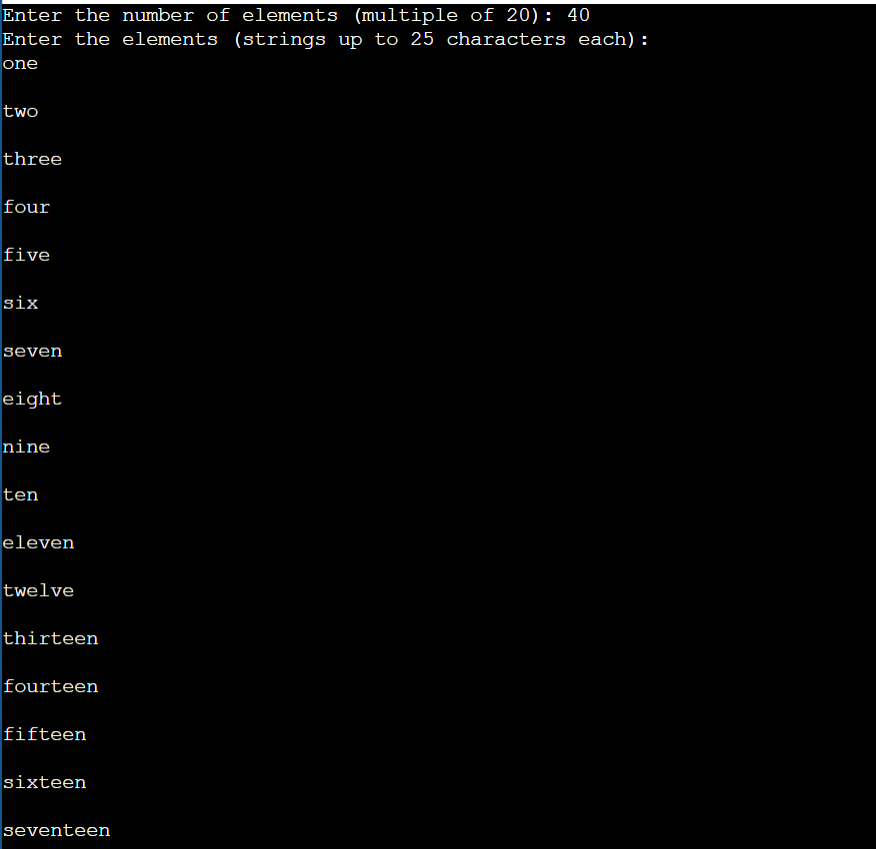
    free\_list(new\_list);

    return EXIT\_SUCCESS;

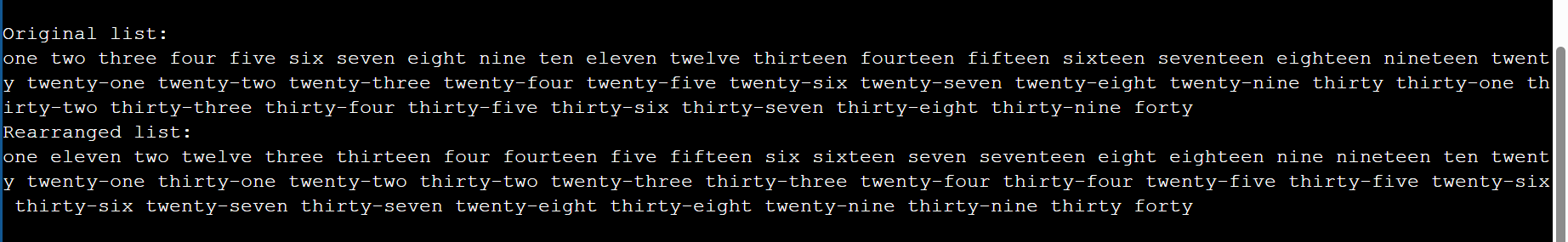
}

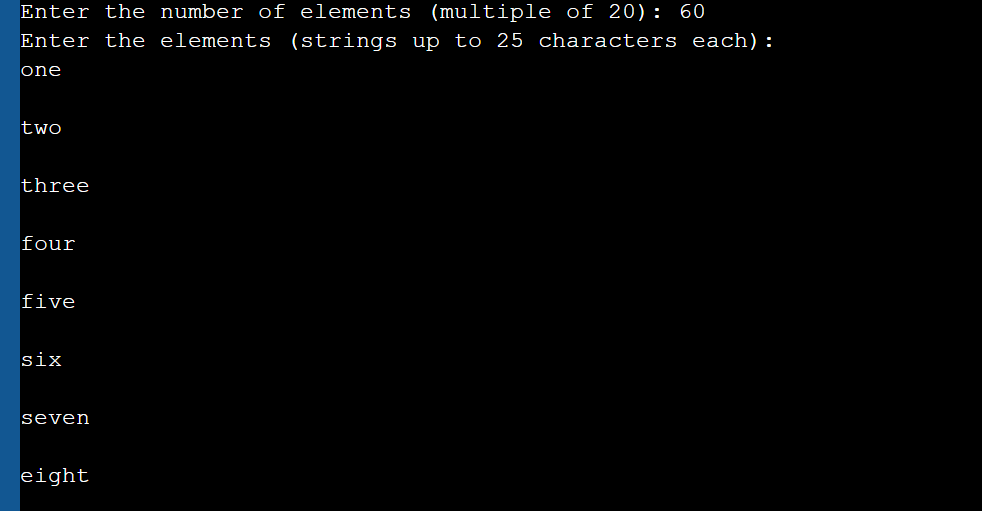
**Результати тестування:  
**



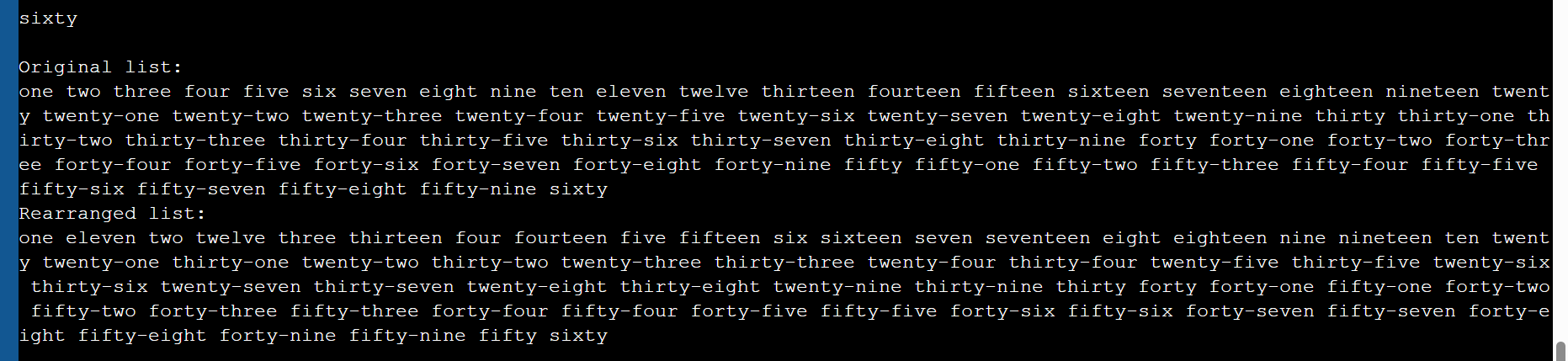


|  |  |
| --- | --- |
|  |  |





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





**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.