

Тема: Динамічні масиви.

Мета: Придбати практичні навички з використанням динамічних масивів.

### Індивідуальне завдання

- розробити функцію, яка реалізує вставку в рядок "s" другий рядок "s2" в "i"-у позицію рядка "s". Наприклад, `insert("abrakadabra", "TEXT2", 4)` повинна створити рядок "abraTEXT2kadabra";
- розробити функцію видалення з рядка "s" усіх символів з індексами в заданому діапазоні. Наприклад, `reduce("abrakadabra", 4, 8)` повинна створити рядок "abrara" (без підрядка kadab).
- за допомогою функцій `memset`, `memset` створити функції додання та видалення елементів з динамічного масиву вашої прикладної області

### Хід роботи

```
int main() {  
    char* mass = (char*)malloc(20 * sizeof(char));  
    strcat(mass, "abrakadabra");  
    char mass2[6] = "TEXT2";  
    mass = insert(mass, mass2, 4);  
    reduce(mass, 4, 8);  
    return 0;  
}  
  
char* insert(char* mass, char* mass2, int index) { ... }  
  
void reduce(char *mass, int indexStart, int indexEnd) { ... }
```

Рис.1 - Фрагмент коду

```

char* insert(char* mass, char* mass2, int index) {

    int size = strlen(mass) + strlen(mass2);
    char* s = (char*)malloc((size + 1) * sizeof(char));

    for (int i = 0; i < index; i++)
        s[i] = mass[i];
    for (int i = index; i < strlen(mass2) + index; i++)
        s[i] = mass2[i - index];
    for (int i = strlen(mass2) + index; i < size; i++)
        s[i] = mass[i - strlen(mass2)];

    s[size] = '\0';

    printf("%s\n", s);

    return s;
}

```

Рис.2 - функція вставки тексту.

```

void reduce(char* mass, int indexStart, int indexEnd) {

    int size = strlen(mass) - (indexEnd - indexStart + 1);
    char* s = (char*)malloc((size + 1) * sizeof(char));

    for (int i = 0; i < indexStart; i++)
        s[i] = mass[i];

    for (int i = indexEnd + 1; i < strlen(mass); i++)
        s[i - (indexEnd - indexStart + 1)] = mass[i];
    s[size] = '\0';
    printf("%s", s);
    return;
}

```

Рис.3 - функція видалення тексту.

```

Консоль отладки Microsoft Visual Studio
abraTEXT2kadabra
abrakadabra

```

Результат виконання програми

```

int addEmployee(struct Employee* employee, int size) {
    size++;
    Employee* mass = (Employee*)malloc(size * sizeof(Employee));
    memcpy(mass, employee, sizeof(Employee) * (size - 1));
    mass[size - 1] = create();
    memcpy(employee, mass, sizeof(Employee) * size);

    return size;
}

```

Рис.4 - функція додавання елементів.

```

int removeEmployee(struct Employee* employee, int size, int index) {
    size--;
    Employee* mass = (Employee*)malloc(size * sizeof(Employee));
    for (int i = 0; i < size; i++)
    {
        if (i < index)
        {
            mass[i] = employee[i];
        }
        else
        {
            mass[i] = employee[i+1];
        }
    }
    memcpy(employee, mass, sizeof(Employee) * size);

    return size;
}

```

Рис.4 - функція видалення елемента.

```

void testAdd() {

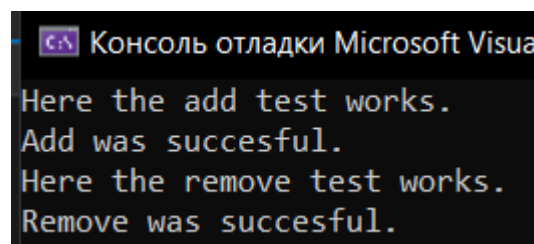
    printf("Here the add test works.\n");

    int size = 2;
    Employee* mass = (Employee*)malloc(size * sizeof(Employee));
    for (int i = 0; i < size; i++)
    {
        mass[i] = create();
    }
    size = addEmployee(mass, size);

    if (mass[size - 1].workExperience > 0 && size == 3)
    {
        printf("Add was succesful.\n");
    }
    else
    {
        printf("Add was not succesful.\n");
    }
}

```

Рис.5 - приклад тесту для додавання.



```

Консоль отладки Microsoft Visual Studio
Here the add test works.
Add was succesful.
Here the remove test works.
Remove was succesful.

```

Результат тестування функцій наведених вище.

**Висновок:** в лабораторній роботі отримані навички з роботою динамічних рядків, а також навчились використовувати memset, memsru для додавання та видалення елементів з динамічного масиву прикладної області Employee