**МIНIСТЕРСТВО  ОСВIТИ І НАУКИ  УКРАЇНИ**

**НАЦІОНАЛЬНИЙ   ТЕХНІЧНИЙ УНІВЕРСИТЕТ   УКРАЇНИ**

**“КИЇВСЬКИЙ  ПОЛІТЕХНІЧНИЙ  ІНСТИТУТ**

**ІМЕНІ ІГОРЯ СІКОРСЬКОГО”**

**Факультет прикладної математики**

**Кафедра програмного забезпечення комп’ютерних систем**

**Лабораторна робота № 1**

**з дисципліни “ Основи програмування ”**

**тема “Файлові потоки”**

|  |  |  |
| --- | --- | --- |
| **Виконав**  **студент I курсу**  **групи КП-82**  **Мельничук Олексій Геннадійович**  **варіант №15** |  | **Перевірив**  **“\_\_\_\_” “\_\_\_\_\_\_\_\_\_\_\_\_” 20\_\_\_ р.**  **викладач**  **Гадиняк Руслан Анатолійович**  **(прізвище, ім’я, по батькові)** |

**Київ 2019**

**Мета роботи**

Навчитись створювати модулі в мові С.

Навчитись реалізовувати основні колекції елементів різними способами.

Навчитись розбивати код проекту на декілька файлів і компілювати його.

**Постановка завдання**

Загальні вимоги до завдання

Створити у репозиторії progbase2 директорію проекта labs/lab1/

Додати у директорію проекта текстовий файл data.txt, що містить довільні рядки з англійськими словами та реченнями. Кожне речення з нового рядка.

Реалізувати список рядків на основі динамічного масиву.

Зчитати всі текстові рядки з текстового файлу і додати їх у список.

Вивести вміст списку у консоль.

Перенести всі рядки, у яких немає слова "the", на початок списку.

Вивести вміст списку у консоль.

Реалізувати стек рядків на основі динамічного масиву

Створити два екземпляри даного типу даних.

Переписати всі елементи непарних позицій списку у перший стек, а всіх парних - у другий.

Вивести вміст обох екземплярів у консоль.

Створити новий список і переписати у нього всі елементи з першого стека, а потім з другого.

Вивести вміст списку у консоль.

Весь код повинен бути розбитим на модулі відповідно до завдання:

main - головний модуль з функцією main()

list - модуль списку

stack - модуль даного АТД

При найменуванні модулів, типів та операцій обов'язково слідувати рекомендаціям.

Кожен модуль розміщувати у двох файлах (.h i .c). Компілювати проект за допомогою команди (з директорії проекта):

gcc \*.c

При створенні модулів допускається використання відкритих структур, але зовнішній код не повинен напряму доступатися до полів такої структури (перевіряється тимчасовим перенесенням відкритої структури у реалізацію, при чому єдиною помилкою має стати створення екземпляра на стеку).

**Тексти коду програм**

|  |
| --- |
| **main.c** |
| #include "list.h"  #include "stack.h"  #include "addon.h"  int main(int argc, char \*argv[argc])  {  List dlist;  List\_init(&dlist);  List\_fwrite(&dlist);  List\_print(&dlist);  List\_sort(&dlist);  List\_print(&dlist);  Stack stack1;  Stack\_init(&stack1);  Stack stack2;  Stack\_init(&stack2);  List dlist2;  List\_init(&dlist2);  Write\_to\_stacks(&stack1, &stack2, &dlist);  Stack\_print(&stack1);  Stack\_print(&stack2);  Write\_to\_list(&stack1, &dlist2);  Write\_to\_list(&stack2, &dlist2);  List\_print(&dlist2);  List\_deinit(&dlist);  List\_deinit(&dlist2);  Stack\_deinit(&stack1);  Stack\_deinit(&stack2);  printf("Fin.\n");  return 0;  } |

|  |
| --- |
| **list.h** |
| #pragma once  #include <stdbool.h>  #include <stdio.h>  #include <string.h>  #include <malloc.h>  #include <stdlib.h>  // List module interface  typedef struct \_\_List List;  struct \_\_List  {  char \*\*array;  int capacity;  int length;  };  void List\_init(List \*self);  void List\_deinit(List \*self);  void List\_add(List \*self, char \*string, int len);  void List\_print(const List \*self);  void List\_fwrite(List \*self);  void List\_sort(List \*self); |

|  |
| --- |
| **list.c** |
| // #include <stdio.h>  // #include <stdlib.h>  // #include <string.h>  // #include <assert.h>  // #include <stdbool.h>  #include "list.h"  void List\_init(List \*self)  {  self->length = 0;  self->capacity = 2;  self->array = malloc(self->capacity \* sizeof(char\*));  if (self->array == NULL)  {  fprintf(stderr, "Allocation error");  abort();  }  }  void List\_deinit(List \*self)  {  for(int i = 0; i < self->length; i++)  {  free(self->array[i]);  }  free(self->array);  }  static void List\_realloc(List \*self, int newCapacity)  {  char \*\*newArray = realloc((self->array), sizeof(char\*) \* newCapacity);  if (newArray == NULL)  {  free(self->array);  fprintf(stderr, "Reallocation error");  abort();  }  self->array = newArray;  self->capacity = newCapacity;  // printf("Reallocated\n");  }  void List\_add(List \*self, char \*string, int len)  {  self->array[self->length] = malloc((len + 1) \* sizeof(char));  strcpy(self->array[self->length], string);  //  self->length += 1;  //  if (self->length == self->capacity)  {  // realloc  int newCapacity = self->capacity + 1;  List\_realloc(self, newCapacity);  }  }  void List\_print(const List \*self)  {  printf("Printing list: \n\n");  for(int i = 0; i < self->length; i++)  {  // printf("%d\n", (int)strlen(self->array[i]));  printf("%s\n", self->array[i]);  }  puts("");  }  void List\_sort(List \*self)  {  printf("Sorting...\n");  char \*bufpoint;  for(int i = 0; i < self->length; i++)  {  if((strstr(self->array[i], " the ") != NULL))  {  bufpoint = self->array[i];  for(int j = i; j < self->length; j++)  {  if((strstr(self->array[j], " the ") == NULL))  {  self->array[i] = self->array[j];  self->array[j] = bufpoint;  break;  }  }  }  }  }  void List\_fwrite(List \*self)  {  FILE \*file;  char buf[256] = {'\0'};  int bufc = 0;  file = fopen("data.txt", "r");  while (fgets(buf, 256, file))  {  while (buf[bufc] != '\0')  {  if (buf[bufc] == '\n') buf[bufc] = '\0';  bufc++;  }  List\_add(self, buf, bufc);  for(int i = 0; i < bufc; i++)  {  buf[i] = '\0';  }  bufc = 0;  }  fclose(file);  } |

|  |
| --- |
| **stack.h** |
| #pragma once  #include <stdbool.h>  #include <stdio.h>  #include <string.h>  #include <malloc.h>  #include <stdlib.h>  typedef struct \_\_Stack Stack;  struct \_\_Stack  {  char \*\*array;  int capacity;  int length;  };  void Stack\_init(Stack \*self);  void Stack\_deinit(Stack \*self);  void Stack\_push(Stack \*self, char \*string, int len);  char \* Stack\_pop(Stack \*self);  char \* Stack\_peek(Stack \*self);  bool Stack\_isEmpty(Stack \*self);  //  void Stack\_print(const Stack \*self); |

|  |
| --- |
| **stack.c** |
| #include <stdio.h>  #include <math.h>  #include <stdlib.h>  #include <time.h>  #include <progbase.h>  #include <progbase/console.h>  #include <stdbool.h>  #include <ctype.h>  #include <string.h>  #include <malloc.h>  #include "stack.h"  // Stack module implementation  void Stack\_init(Stack \*self)  {  self->length = 0;  self->capacity = 4;  self->array = malloc(self->capacity \* sizeof(char\*));  if (self->array == NULL)  {  fprintf(stderr, "Allocation error");  abort();  }  }  void Stack\_deinit(Stack \*self)  {  for(int i = 0; i < self->length; i++)  {  free(self->array[i]);  }  free(self->array);  }  static void Stack\_realloc(Stack \*self, int newCapacity)  {  char \*\*newArray = realloc(self->array, sizeof(char\*) \* newCapacity);  if (newArray == NULL)  {  for (int i = 0; i < self->length; i++)  {  free(self->array[i]);  }  free(self->array);  fprintf(stderr, "Reallocation error");  abort();  }  self->array = newArray;  self->capacity = newCapacity;  // printf("Reallocated");  }  void Stack\_push(Stack \*self, char \*string, int len)  {  self->array[self->length] = malloc((len + 1) \* sizeof(char));  strcpy(self->array[self->length], string);  self->length += 1;  //  if (self->length == self->capacity)  {  // realloc  int newCapacity = self->capacity + 1;  Stack\_realloc(self, newCapacity);  }  }  char \* Stack\_pop(Stack \*self)  {  if (self->length == 0)  {  fprintf(stderr, "Stack underflow");  abort();  }  char \*string = self->array[self->length - 1];  self->length -= 1;  return string;  }  char \* Stack\_peek(Stack \*self)  {  if (self->length == 0)  {  fprintf(stderr, "Stack underflow");  abort();  }  char \*string = self->array[self->length - 1];  return string;  }  bool Stack\_isEmpty(Stack \*self)  {  return self->length == 0;  }  void Stack\_print(const Stack \*self)  {  printf("Printing stack: \n\n");  for(int i = 0; i < self->length; i++)  {  // printf("%d\n", (int)strlen(self->array[i]));  printf("%s\n", self->array[i]);  }  puts("");  } |

|  |
| --- |
| **addon.h** |
| #include <stdbool.h>  #include <stdio.h>  #include <string.h>  #include "stack.h"  #include "list.h"  void Write\_to\_stacks(Stack \*self1, Stack \*self2, List \*self);  void Write\_to\_list(Stack \*stacks, List \*dlist); |

|  |
| --- |
| **addon.c** |
| #include "addon.h"  void Write\_to\_stacks(Stack \*self1, Stack \*self2, List \*self)  {  printf("Writing into stacks: \n");  for(int i = 0; i < self->length; i++)  {  if(i % 2 == 0)  {  Stack\_push(self1, self->array[i], (int)strlen(self->array[i]));  }  else if(i % 2 != 0)  {  Stack\_push(self2, self->array[i], (int)strlen(self->array[i]));  }  }  }  void Write\_to\_list(Stack \*stack, List \*dlist)  {  printf("Writing into list..\n");  for(int i = 0; i < stack->length; i++)  {  List\_add(dlist, stack->array[i], (int)strlen(stack->array[i]));  }  } |

|  |
| --- |
| **data.txt** |
| As I walk through the valley of the shadow of death  I take a look at my life and realize there's nothin' left  ‘Cause I've been blastin' and laughin' so long  That even my momma thinks that my mind is gone  But I ain't never crossed a man that didn't deserve it  Me be treated like a punk, you know that's unheard of  You better watch how you talkin' and where you walkin'  Or you and your homies might be lined in chalk  I really hate to trip, but I gotta loc  As they croak, I see myself in the pistol smoke  Fool, I'm the kinda G the little homies wanna be like  On my knees in the night, sayin' prayers in the streetlight  Been spendin' most their lives  Livin' in the gangsta's paradise  Been spendin' most their lives  Livin' in the gangsta's paradise  Keep spendin' most our lives  Livin' in the gangsta's paradise  Keep spendin' most our lives  Livin' in the gangsta's paradise |

**Робота коду**

Створюються усі потрібні структури і функції.

Реалізується виділення пам’яті для масиву вказівників на строки.

Відкривається файловий потік.

Текстовий файл зчитується та додається у буфер, знаходиться розмір строки, виділяється пам’ять для одної строки списку, вміст буфера передається у строку та очищується.

Файловий потік закривається.

Список сортується відповідно до умови шляхом переставляння вказівників.

Створюються два стеки, виділяється пам’ять.

Вміст списку передається у стеки відповідно до умови, виділяється пам’ять до кожної строки у стеках.

Створюється новий список, виділяється пам’ять.

Вміст кожного стека по черзі наповнює новий список, виділяється пам’ять.

Усі списки і стеки деініціалізуються, очищується пам’ять.

Код розбивається на інтерфейс та реалізації у окремі header-файли і зв’язується.

**Приклади результатів**

Результат у консолі:

|  |
| --- |
| aleksey@auv:~/Desktop/local code/VARIOUS/test/LB1$ gcc \*.c  aleksey@auv:~/Desktop/local code/VARIOUS/test/LB1$ ./a.out  Printing list:  As I walk through the valley of the shadow of death  I take a look at my life and realize there's nothin' left  ‘Cause I've been blastin' and laughin' so long  That even my momma thinks that my mind is gone  But I ain't never crossed a man that didn't deserve it  Me be treated like a punk, you know that's unheard of  You better watch how you talkin' and where you walkin'  Or you and your homies might be lined in chalk  I really hate to trip, but I gotta loc  As they croak, I see myself in the pistol smoke  Fool, I'm the kinda G the little homies wanna be like  On my knees in the night, sayin' prayers in the streetlight  Been spendin' most their lives  Livin' in the gangsta's paradise  Been spendin' most their lives  Livin' in the gangsta's paradise  Keep spendin' most our lives  Livin' in the gangsta's paradise  Keep spendin' most our lives  Livin' in the gangsta's paradise  Sorting...  Printing list:  I take a look at my life and realize there's nothin' left  ‘Cause I've been blastin' and laughin' so long  That even my momma thinks that my mind is gone  But I ain't never crossed a man that didn't deserve it  Me be treated like a punk, you know that's unheard of  You better watch how you talkin' and where you walkin'  Or you and your homies might be lined in chalk  I really hate to trip, but I gotta loc  Been spendin' most their lives  Been spendin' most their lives  Keep spendin' most our lives  Keep spendin' most our lives  As I walk through the valley of the shadow of death  Livin' in the gangsta's paradise  As they croak, I see myself in the pistol smoke  Livin' in the gangsta's paradise  Fool, I'm the kinda G the little homies wanna be like  Livin' in the gangsta's paradise  On my knees in the night, sayin' prayers in the streetlight  Livin' in the gangsta's paradise  Writing into stacks:  Printing stack:  I take a look at my life and realize there's nothin' left  That even my momma thinks that my mind is gone  Me be treated like a punk, you know that's unheard of  Or you and your homies might be lined in chalk  Been spendin' most their lives  Keep spendin' most our lives  As I walk through the valley of the shadow of death  As they croak, I see myself in the pistol smoke  Fool, I'm the kinda G the little homies wanna be like  On my knees in the night, sayin' prayers in the streetlight  Printing stack:  ‘Cause I've been blastin' and laughin' so long  But I ain't never crossed a man that didn't deserve it  You better watch how you talkin' and where you walkin'  I really hate to trip, but I gotta loc  Been spendin' most their lives  Keep spendin' most our lives  Livin' in the gangsta's paradise  Livin' in the gangsta's paradise  Livin' in the gangsta's paradise  Livin' in the gangsta's paradise  Writing into list..  Writing into list..  Printing list:  I take a look at my life and realize there's nothin' left  That even my momma thinks that my mind is gone  Me be treated like a punk, you know that's unheard of  Or you and your homies might be lined in chalk  Been spendin' most their lives  Keep spendin' most our lives  As I walk through the valley of the shadow of death  As they croak, I see myself in the pistol smoke  Fool, I'm the kinda G the little homies wanna be like  On my knees in the night, sayin' prayers in the streetlight  ‘Cause I've been blastin' and laughin' so long  But I ain't never crossed a man that didn't deserve it  You better watch how you talkin' and where you walkin'  I really hate to trip, but I gotta loc  Been spendin' most their lives  Keep spendin' most our lives  Livin' in the gangsta's paradise  Livin' in the gangsta's paradise  Livin' in the gangsta's paradise  Livin' in the gangsta's paradise  Fin.  aleksey@auv:~/Desktop/local code/VARIOUS/test/LB1$ |

**Висновки**

Дана лабораторна робота допомогла розібратися з динамічним виділенням пам’яті, а також у розумінні абстрактних структур даних, таких як стек.

Також був охоплений стиль модульного програмування, розбиття коду на файли та лінкування.

У роботі були використані стандартні бібліотеки мови C, valgrind для перевірки на проблеми з пам’яттю а також компілятор gcc.