Звіт з лабораторної роботи
за дисципліною "програмування"
студента групи ПА-17-1
Панасенка Егора Сергійовича
Кафедра комп’ютерних технологій, фпм, дну
2017/2018 навч.р.

1. Постановка задачі: Написати програму, у якій були би розроблені 2 функції, імітуючі дії стандартних строкових функцій (strcpy та memove). Вибір функцій здійснюються за допомогою пунктів меню.
2. Опис ходу розв’язку:
    1. Задається функція stringcopy, яка імітує strcpy, але працює повільніше так яка частина коду в strcpy на асемблері. У функції за допомогою адресу відбувається ітерація і присвоювання і виводитися адреса першого елемента масиву.
    2. Задається функція memorymove, яка імітує memmove, працює схоже як і функція stringcopy, але переноситься тільки частина блоку пам'яті і ітерація може відбуватися у зворотньому напрямку, якщо блоки перетинаються.
    3. Задається функція ign_other, у якій будуть ігноруватися усі данні до кінця рядка для того щоб непотрібні дані не заважали в майбутньому.
    4. Задається функція get_string, яка забирає рядок з консолі фбо з файлу та зберігає його у масив, а розмір рядка обмежується максимально доступним значенням int, а також розмір збільшується частинами рівними CHUNK (128).
    5. У головній функції виконуються такі дії:
        1. Виконується меню, доки не отримаємо потрібну відповідь, що робити далі
        2. Запитується потрібний рядок.
        3. Далі якщо вибрали у меню stringcopy:
            1. Задаються два рядки схожого роміру (string1 і string2)
            2. Виконується копіювання введеного рядка у string1, а далі з string1 у string2 за допомогою stringcopy.
            3. Замірюється час виконання stringcopy.
            4. Виконується копіювання введеного рядка у string1, а далі з string1 у string2 за допомогою strcpy.

5. Замірюється час виконання strcpy.
6. Виводиться відношення швидкостей.

4. Якщо вибрали у меню memmove:
1. Дублюється введений рядок у змінну b.
2. Запитуються потрібні данні для виконання переносу блока.
3. Виконується memorymove для введеного рядка
4. Виконується memmove для b.
5. Виводиться відношення швидкостей.

3. Вихідний текст програми розв’язку задачі

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <time.h>
#include <limits.h>
#define CHUNK 128

/* Copy string from source (second argument) to destanation (first argument) */
char *stringcopy(char *d, const char *s) {
	// Saving begining of destanation string
	char *saved = d;
	// While current character is not zero copy each
	// character from source to destanation string
	while (*s)
		// Same as *d = *s;d++;s++;
		// Copy character from s to d
		// then increase d and s
		*d++ = *s++;
	*d = 0; // Make end of string by zero
	return saved; // Returning destanation string
}

/* Move block of memory from source (second argument)
 * to destanation (first argument) with size of third argument */
void *memorymove(void *dest, const void *src, long c) {
	// Create dynamic pointers
	char * d = (char*) dest;
	const char * s = (const char*) src;
	// Check that memory block not overlaps
	// and if they will overlap copy blocks in reverse direction
	if (d - s <= c) {
		// While count is not zero decrease counter and
		// copy character from source to destanation block
		while (c--)
			*d++ = *s++;
		return dest;
	} else {
		// Add to each pointers count number
		d+=c;s+=c;
		// While count is not zero decrease counter and
		// copy character from source to destanation block
		while (c--)
```

```
                *d-- = *s--;
            return dest;
      }
}

void ign_other(FILE * input) {
      char c = 0;
      while ((c = fgetc(input)) != EOF && c != '\n') {}
}

char * get_string (FILE * input, char * status, int * len) {
      char * a = (char *) calloc(CHUNK,sizeof(char)), ch = EOF;
      int length = 0;
      while (ch) {
            ch = fgetc(input);
            if (length >= INT_MAX - 2 * CHUNK || ch == EOF || ch == '\n') {
                  if (ch == EOF)
                        *status = 2;
                  else if (ch == '\n')
                        *status = 3;
                  else if (length >= INT_MAX - 2 * CHUNK)
                        *status = 1;
                  ch = 0;
            }
            if (length % CHUNK == CHUNK - 1)
                  a = (char *) realloc (a, (length + CHUNK + 1) * sizeof(char));
            a[length++] = ch;
      }
      *len = length;
      return a;
}

int main() {
      // Debbuging clock
      clock_t start, end;
      double cpu_time_used1,cpu_time_used2;
      char answer = 0, *a, status = 0;
      int length = 0;
      // While answer is incorrect make menu
      while (answer < 1 || answer > 2) {
            // Ask menu
            printf("Выберете фукцию:\n1. strcpy()\n2. memmove()\n3. exit()\n");
            // Get answer
            answer = getchar()-'0';
            // Ignore all after answer to avoid incorrect data in future
            ign_other(stdin);
            // Exit if answer is 3
            if (answer == 3) exit(0);
            // Else ask to try again
            else if (answer < 1 || answer > 2) {
                  system("clear");
                  printf("Вводить нужно только цифры от 1 до 3. Попробуйте ещё\n");
            }
      }
      while (status < 2) {
            printf("Введите строку: ");
            a = get_string(stdin,&status,&length);
            if (status == 1) {
```

```
                  system("clear");
                  printf("Слишком длинная строчка, максимальная длина строчки %i.
Попробуйте ещё.\n", INT_MAX - 2 * CHUNK);
                  ign_other(stdin);
            }
            if (status < 2) free(a);
      }
      // Make something by answer
      if (answer == 1) {
            // Create strings with certain length, to avoid overflows
            char string1[length],string2[length];
            // Saving starting time
            start = clock();
            printf("Воспользуемся самодельными функциями\n");
            // Copying a to string1
            printf("Копируем введенную строчку в string1\n");
            stringcopy(string1,a);
            // Copying string1 to string2
            printf("Копируем string1 в string2\n");
            stringcopy(string2,string1);
            printf("string1: %s\nstring2: %s\n",string1,string2);
            // Saving ending time
            end = clock();
            // Get execution time
            cpu_time_used1 = ((double) (end - start));
            printf("Операции выполнились за %f милисекунд\n",cpu_time_used1);
            // Saving starting time
            start = clock();
            printf("Воспользуемся стандартными функциями\n");
            // Copying a to string1
            printf("Копируем введенную строчку в string1\n");
            strcpy(string1,a);
            // Copying string1 to string2
            printf("Копируем string1 в string2\n");
            strcpy(string2,string1);
            printf("string1: %s\nstring2: %s\n",string1,string2);
            // Saving ending time
            end = clock();
            // Get execution time
            cpu_time_used2 = ((double) (end - start));
            // Show debug data
            printf("Операции выполнились за %f милисекунд\n",cpu_time_used2);
            printf("Производительность уменьшилась в
%f\n",cpu_time_used1/cpu_time_used2);
      } else {
            int x=0,y=0,l=0;
            char b[length];
            strcpy(b,a);
            while (x == y || x+l > length || y+l > length || length == 0) {
                  // Input destanation memory block position
                  printf("Введите позицию блока которую хотите заменить: ");
                  scanf("%i",&x);
                  // Input source memory block position
                  printf("Введите позицию блока с помощью которой хотите заменить:
");
                  scanf("%i",&y);
                  // Input length of memory block
                  printf("Введите длину блока: ");
```

```
                scanf("%i",&l);
                // Warn on incorrect input data
                if (x == y || x+l > length || y+l > length || length == 0) {
                        system("clear");
                        if (x == y || length == 0)
                                printf("Ничего не поменяеться. ");
                        else if (x+l > length || y+l > length)
                                printf("Блок памяти вылазит за строку");
                        printf("Попробуйте ещё\n");
                }
        }
        // Saving starting time
        start = clock();
        printf("Воспользуемся самодельными функциями\n");
        // Move memory block from x to y
        printf("Переместим часть строки\n");
        memorymove(a+x-1,a+y-1,l);
        printf("Получившиеся строка: %s\n",a);
        // Saving ending time
        end = clock();
        // Get execution time
        cpu_time_used1 = ((double) (end - start));
        printf("Операции выполнились за %f милисекунд\n",cpu_time_used1);
        // Saving starting time
        start = clock();
        printf("Воспользуемся самодельными функциями\n");
        // Move memory block from x to y
        printf("Переместим часть строки\n");
        memmove(b+x-1,b+y-1,l);
        printf("Получившиеся строка: %s\n",b);
        // Saving ending time
        end = clock();
        // Get execution time
        cpu_time_used2 = ((double) (end - start));
        // Show debug data
        printf("Операции выполнились за %f милисекунд\n",cpu_time_used2);
        printf("Производительность уменьшилась в
%f\n",cpu_time_used1/cpu_time_used2);
    }
    // Exit
    free(a);
    return 0;
}
```

4. Опис інтерфейсу програми:
    1. Вибір функції яку ми бажаємо перевірити або вихід
    2. Вхід будь-якого рядка.
    3. Якщо вибрати в пункті 1 strcpy, то виконається така послідовність дій:
        1. Рядок який ми внесли копіюється у string1, а string1 в string2
        2. виконується це за допомогою стандартних функцій а також за допомогою функцій зроблених власноручно і перевіряється швидкість виконання обох способів і виводиться відношення

швидкостей.

4. Якщо вибрати в пункті 1 memmove, то виконається така послідовність дій:
   1. Рядок який ввели дублюється.
   2. Виконується запит даних необхідних для роботи програми.
   3. І виконується послідовність дій як у пункті 3.2
5. Опис тестових прикладів:

```
Выберете фукцию:
1. strcpy()
2. memmove()
3. exit()
1
Введите строку: asdfghj
Воспользуемся самодельными функциями
Копируем введенную строчку в string1
Копируем string1 в string2
string1: asdfghj
string2: asdfghj
Операции выполнились за 72.000000 милисекунд
Воспользуемся стандартными функциями
Копируем введенную строчку в string1
Копируем string1 в string2
string1: asdfghj
string2: asdfghj
Операции выполнились за 37.000000 милисекунд
Производительность уменьшилась в 1.945946


Выберете фукцию:
1. strcpy()
2. memmove()
3. exit()
2
Введите строку: asdfgh
Введите позицию блока которую хотите заменить: 2
Введите позицию блока с помощью которой хотите заменить: 3
Введите длину блока: 2
Воспользуемся самодельными функциями
Переместим часть строки
Получившиеся строка: adffgh
Операции выполнились за 123.000000 милисекунд
Воспользуемся самодельными функциями
Переместим часть строки
Получившиеся строка: adffgh
Операции выполнились за 58.000000 милисекунд
Производительность уменьшилась в 2.120690
```