|  |  |
| --- | --- |
|  | **Министерство науки и высшего образования Российской Федерации**  **Мытищинский филиал**  **Федеральное государственное бюджетное образовательное учреждение**  **высшего образования**  **«Московский государственный технический университет**  **имени Н.Э. Баумана**  **(национальный исследовательский университет)»**  **(МГТУ им. Н.Э. Баумана)** |

ФАКУЛЬТЕТ Космический

КАФЕДРА «Прикладная математика, информатика и вычислительная техника» К3-МФ

**Лабораторная работа**

*ПО ДИСЦИПЛИНЕ:*

***Системное программное обеспечение***

***НА ТЕМУ:***

***\_\_\_\_\_\_\_\_\_Создание простого макропроцессора\_\_\_\_\_\_\_\_***

***\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_для языка Ассемблера \_\_\_\_\_\_\_\_\_\_\_\_\_\_***

***\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_***

***\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_***

Студент \_\_К3-53Б\_\_  **\_\_\_\_\_\_\_\_\_\_** Цветков Юрий Алексеевич

(Группа) (Подпись, дата) (И.О.Фамилия)

#### Преподаватель \_\_\_\_\_\_\_\_\_\_\_\_ Чернышов Александр Викторович

(Подпись, дата) (И.О.Фамилия)

*2021 г.*

**Задание**

Написать на языке программирования Си программу простого макропроцессора для любого языка Ассемблера. Программа макропроцессора должна поддерживать следующие элементы:

* обнаруживать и запоминать макроопределения (текстовые строки между директивами macro и endm);
* запоминать имена макроопределений;
* распознавать формальные позиционные макропараметры;
* по именам макроопределений обнаруживать вызовы макрогенерации

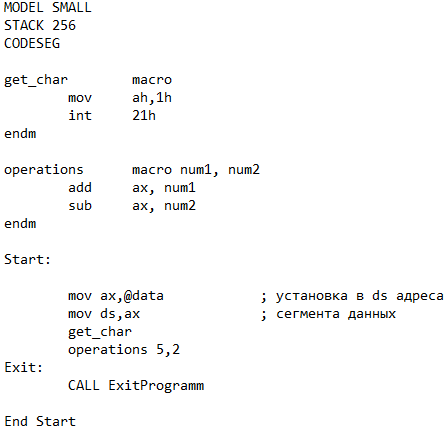
в тексте программы на Ассемблере:

* распознавать фактические позиционные макропараметры;
* выполнять макроподстановку с учётом распознанных формальных и фактических макропараметров.

***Код main.c***

#include <stdio.h>  
#include <stdlib.h>  
#include <string.h>  
#define **size** 32  
  
struct macro;  
typedef struct macro\* pMacro;  
typedef struct macro\*\* ppMacro;  
  
struct macro  
{  
 int num; *//количество макропараметров* char name[**size**]; *// имя макроопределния* char param[**size**][**size**]; *//макропараметры* char code[**size**\***size**]; *// макроопределене*};  
  
void macroAdds(char\* file1, int\* num\_macro, ppMacro macro) *//метод заполняет структуры макросов*{  
 FILE\* in = fopen(file1, "r");  
  
 char temp[**size**\***size**];  
  
 while(fgets(temp, **size**\***size**, in) != **NULL**)  
 {  
 char\* str = strstr(temp, "macro");  
 if(str != **NULL**)  
 {  
 macro[\*num\_macro] = (pMacro)malloc(sizeof(struct macro));  
 macro[\*num\_macro] -> num = 0;  
 strncpy(macro[\*num\_macro] -> name, temp, strlen(temp) - strlen(str) - 1); *//присваиваем имя* char\* params = str + strlen("macro");  
 if(strcmp("\n", params) != 0) *//проверяем есть ли параметры* {  
 char\* param = strtok(params," ,\n");  
 for(int i = 0; param != **NULL**; i++) *//записываем в структуру все параметры* {  
 strcpy(macro[\*num\_macro] -> param[i], param);  
 macro[\*num\_macro] ->num++;  
 param = strtok(**NULL**, " ,\n");  
 }  
 }  
 memset(macro[\*num\_macro] ->code, '\0', **size**\***size**);  
 while(fgets(temp, **size**\***size**, in) != **NULL**) *// заполняем поле с макроопределением* {  
 if(strstr(temp, "endm") == **NULL**)  
 strcat(macro[\*num\_macro] -> code, temp);  
 else  
 {  
 \*num\_macro+=1;  
 break;  
 }  
 }  
 }  
 }  
 fclose(in);  
}  
  
void replace(char\* str, char\* substr1, char\* substr2)  
{  
 char res[**size**\***size**];  
 memset(res, '\0', **size**\***size**);  
 char\* temp = strstr(str, substr1);  
 while(temp!= **NULL**)  
 {  
 char words[**size**\***size**];  
 memset(words, '\0', **size**\***size**);  
 strncpy(words, str, strlen(str) - strlen(temp));  
 strcat(res, words);  
 strcat(res, substr2);  
 strcat(res, temp + strlen(substr1));  
 strcpy(str, res);  
 temp = strstr(str, substr1);  
 }  
}  
  
void generateCode(char\* file1, char\* file2, int num\_macro, ppMacro macro)  
{  
 FILE\* in = fopen(file1, "r");  
 FILE\* out = fopen(file2, "w");  
  
 char temp[**size**\***size**];  
  
 while(fgets(temp, **size**\***size**, in) != **NULL**)  
 {  
 int pr = 1;  
 for(int i = 0; i < num\_macro; i++)  
 {  
 char\* name = strstr(temp, macro[i] -> name);  
 if(name != **NULL** && strstr(temp, "macro") == **NULL**)  
 {  
 char\* params = name + strlen(macro[i] -> name);  
 char param\_actual[**size**][**size**];  
  
 if(strcmp("\n", params) != 0) *//проверяем есть ли параметры* {  
 char\* param = strtok(params," ,\n");  
 for(int j = 0; param != **NULL**; j++) *//записываем в структуру все параметры* {  
 strcpy(param\_actual[j], param);  
 param = strtok(**NULL**, " ,\n");  
 }  
 }  
  
 char new\_code[**size**\***size**];  
 strcpy(new\_code, macro[i] -> code);  
 for(int j = 0; j < macro[i] -> num; j++) *// замена формальных параметров на фактические* replace(new\_code, macro[i]->param[j], param\_actual[j]);  
  
 fprintf(out, "%s", new\_code);  
 pr = 0;  
 }  
 }  
 if(pr == 1)  
 fprintf(out, "%s", temp);  
 }  
  
 fclose(in);  
 fclose(out);  
}  
  
int main(int argc, char\* argv[])  
{  
 int num\_macro = 0;  
 pMacro macro[**size**];  
 for(int i = 0; i < **size**; i++)  
 macro[i] = **NULL**;  
  
 macroAdds(argv[1], &num\_macro, macro);  
 generateCode(argv[1], argv[2], num\_macro, macro);  
return 0;  
}

**Входной файл**



**Результат работы программы**

