1 практика

1. Цель работы: практическое закрепление знаний об опциях транслятора.
2. Исходный код  
   #include <stdio.h>

#include <math.h>

int main(void){

int iteration = 0; //Чтобы зациклить выполнение программы

double term = 0; //слогаемое

while(iteration++ < 1000000){

double e = 0.0000001; //константа для вычисления (пока слогаемое больше цикл do работает)

//fib values

double fn[2] = {1,1}; //устанавливаем значения fib. на 3 проход(по варианту)

double nextFib = 0; //сюда записывается следующее значение фиб-и

//factorial

double factorial = 1;

//pow

double pow = 5;

double i = 1;

do{

nextFib = fn[0] + fn[1];

fn[0] = fn[1];

fn[1] = nextFib;

factorial = factorial \* i; //i увеличивается каждую итерацию что делат это выражение ф-ом (1\*2\*3\*4...)

pow = pow \* 5;

term = nextFib / (pow \* factorial); //sqrt разрешён вроде

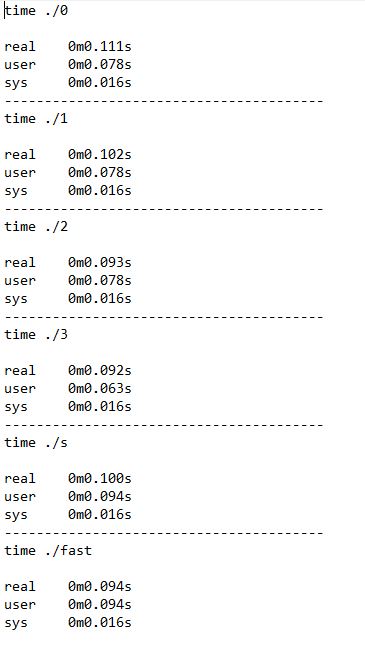
i++;

}while (term > e);

}

}

1. Таблица



2 практика

(a) Цель работы: практическое закрепление знаний об инструментальных средствах, используемых при разработке программ в среде Linux

(b) Исходный текст программы

#include "list.h"

#include <stdio.h>

#include <stdlib.h>

#include <string.h>

char needChar(int mod){

char t[3];

while(t[0] < 49 || t[0] > 55-mod){

fgets(t, 3, stdin);

if (t[0] < 49 || t[0] > 55-mod) printf("Такое значение не подходит попробуйте ещё раз\n");

}

return t[0];

}

void crackEnt(char \*str, int mod){

if(mod == 1){

for (size\_t i = 0; i < 512; i++) {

if (str[i] == '\n') {

str[i] = ';';

strcat(str, " ");

break;

}

}

}

else if(mod == -1){

for (size\_t i = 0; i < 512; i++) {

if (str[i] == '\n') {

str[i] = '\0';

break;

}

}

}

}

void main() {

node head = createNode();

char string1[512];

char string2[512];

char string3[512];

char file[512];

int check = 1;

char inp[512];

while (check) {

printf("Выберете необходимую функцию:\n\n1. Добавить\n2. Удалить\n3. Изменить\n4. Вывести список\n5. Выход\n6. Загрузить из файла\n7. Загрузить в файл\n\n");

//printf("%d((((((sxsxsxsxxsxs\n", needChar());

switch (needChar(0)) {

/\*Добавление эелемента в БД\*/

case '1':

printf("\nВведите название: ");

fgets(string1, 512, stdin);

printf("Выберите тип вещания:\n\n1. обычный\n2. HD\n\n");

switch (needChar(5)) {

case '1':

strcpy(string2, "обычный ");

break;

case '2':

strcpy(string2, "HD ");

break;

}

printf("Выберите флаг общедоступности:\n\n1. Да\n2. Нет\n\n");

switch (needChar(5)) {

case '1':

strcpy(string3, "Да");

break;

case '2':

strcpy(string3, "Нет");

break;

}

for (size\_t i = 0; i < 512; i++) {

if (string1[i] == '\n') {

string1[i] = ' ';

break;

}

}

strcat(string1, string2);

strcat(string1, string3);

printf("\n%s\n\n", string1);

addNode(head, string1);

break;

/\*Удаление\*/

case '2':

{

int i = 0;

printf("Выберите элемент: ");

scanf("%d", &i);

while(i > HowMuchNode(head) || i < 1){

printf("Выберите элемент в доступном диапазоне: ");

scanf("%d", &i);

}

head = DelByIndex(i, head);

}

break;

/\*Изменени;\*/

case '3':

printf("\nВведите номер записи: ");

int x = 0;

scanf("%d", &x);

getchar();

printf("\nВведите название: ");

fgets(string1, 512, stdin);

printf("Выберите тип вещания:\n\n1. обычный\n2. HD\n\n");

switch (needChar(5)) {

case '1':

strcpy(string2, "обычный");

break;

case '2':

strcpy(string2, "HD");

break;

}

printf("Выберите флаг общедоступности:\n\n1. Да\n2. Нет\n\n");

switch (needChar(5)) {

case '1':

strcpy(string3, "Да");

break;

case '2':

strcpy(string3, "Нет");

break;

}

for (size\_t i = 0; i < 512; i++) {

if (string1[i] == '\n') {

string1[i] = ' ';

break;

}

}

for (size\_t i = 0; i < 512; i++) {

if (string2[i] == '\n') {

string2[i] = ' ';

break;

}

}

strcat(string1, string2);

strcat(string1, string3);

printf("\n%s\n\n", string1);

strcpy(NodeByIndex(x, head)->data, string1);

break;

/\*Вывод списка\*/

case '4':

{

node n = head;

n = n->next;

int i = 1;

while (n != NULL) {

printf("%d. %s\n",i++ , n->data);

n = n->next;

}

printf("\n");

}

break;

/\*Закрытие программы\*/

case '5':

return;

break;

case '6':

printf("Введите имя файла без расширения: ");

fgets(file, 512, stdin);

crackEnt(file, -1);

strcat(file, ".bd");

FILE \*fp2 = fopen(file, "rt");

char str[512];

while(fgets(str, 512, fp2) != NULL){

addNode(head, str);

}

fclose(fp2);

printf("\n");

break;

case '7':

printf("Введите имя файла без расширения: ");

fgets(file, 512, stdin);

crackEnt(file, -1);

strcat(file, ".bd");

FILE \*fp = fopen(file, "a+t");

if(head->data != ""){

node tmp = head;

while(tmp->next != NULL){

fputs(tmp->data, fp);

tmp = tmp->next;

}

fputs(tmp->data, fp);

}

fclose(fp);

printf("\n");

break;

}

}

}

(c) Разработанный Makefile

all: main unit runtests clean

main: list.o main.o

gcc -omain list.o main.o

list.o: list.c

gcc -c list.c

main.o: main.c list.h

gcc -c main.c

unit.o: unit.c list.h

gcc -c unit.c

unit: unit.o list.o

gcc -ounit unit.o list.o Unity/src/unity.c

runtests:unit

./unit

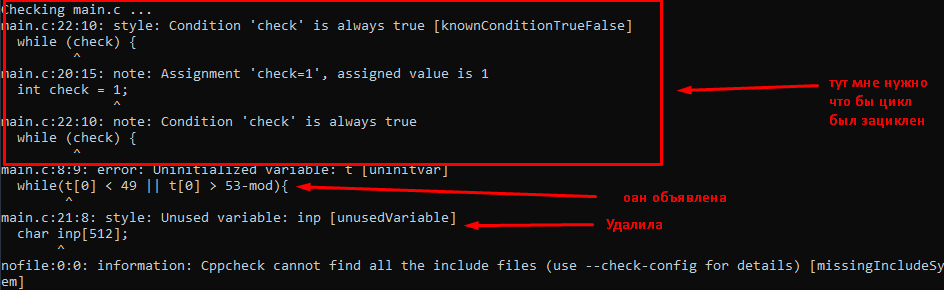
clean:

rm list.o main.o unit.o unit

(d) Протокол git

(f) Скриншоты wiki

(g) Скриншоты результатов работы cppcheck



(h) Исходные тексты UNIT-тестов

#include "Unity/src/unity.h"

#include "list.h"

void testf(){

node test = createNode();

TEST\_ASSERT\_EQUAL\_INT(0, HowMuchNode(test));

addNode(test, "hello world");

addNode(test, "hello world");

TEST\_ASSERT\_EQUAL\_INT(2, HowMuchNode(test));

}

void setUp(/\* arguments \*/) {

/\* code \*/

}

void tearDown(/\* arguments \*/) {

/\* code \*/

}

void main() {

UNITY\_BEGIN();

RUN\_TEST(testf);

return UNITY\_END();

}

(i) Самостоятельно написанные файлы для использования в autotools

# -\*- Autoconf -\*-

# Process this file with autoconf to produce a configure script.

AC\_PREREQ([2.69])

AC\_INIT([bd], [0.01], [google@mail.ru])

AM\_INIT\_AUTOMAKE([foreign -Wall -Werror])

AC\_CONFIG\_SRCDIR([src])

AC\_CONFIG\_HEADERS([config.h])

# Checks for programs.

AC\_PROG\_CC

# Checks for libraries.

# Checks for header files.

AC\_CHECK\_HEADERS([stddef.h stdlib.h string.h])

# Checks for typedefs, structures, and compiler characteristics.

AC\_TYPE\_SIZE\_T

# Checks for library functions.

AC\_FUNC\_MALLOC

AC\_CONFIG\_FILES([Makefile src/Makefile])

AC\_OUTPUT

3 практика

1. Цель работы: знакомство с созданием библиотек в среде Linux
2. исходный тексты, что были изменены в ходе работы (включая Makefile);

binary: main.o libworld.a libworld.so

gcc -o binary main.o -L. -lworld -Wl,-rpath,.

main.o: main.c

gcc -c main.c

libworld.a: list.o

ar cr libworld.a list.o

libworld.so: list.o

gcc -shared -o libworld.so list.o

list.o: list.c

gcc -c -fPIC list.c

clean:

rm -f \*.o \*.a binary

(c) протокол git;

(d) протокол системы отслеживания ошибок (при необходимости);