Miłosz Sieradzki

Dokumentacja generatora drzewa wywołań funkcji w programach w języku C

Spis treści

1. Wstęp

1.1 Opis problemu

Program ma za zadanie generować drzewo wywołań funkcji na podstawie analizy plików z kodem źródłowym zapisanym zgodnie ze standardem C99. Ma być wywoływany z listą argumentów, na które składają się nazwy plików do przeanalizowania (dowolna liczba) oraz przełączniki do następujących opcji:

- drukowanie numerów linii
- ignorowanie wybranych funkcji
- rozpoznawanie makropoleceń

1.2 Narzędzia

Program został napisany w języku C w standardzie C99. Do kompilacji wykorzystano **gcc** w wersji 4.5, do testowania debugger **gdb** w wersji 7.2. Całość zbudowano przy pomocy GNU Make. Dokumentacja została zapisana w formacie PDF.

2. Specyfikacja funkcjonalna

2.1 Sposób użycia

```
drzewo <parametry>
```

2.2 Dopuszczalne parametry

`-m`	informuj o makrodefinicjach
`-n`	wypisuj numery wierszy
`-i`	ignoruj domyślny zestaw funkcji
`-i:lista`	ignoruj zestaw funkcji z pliku lista (patrz też punkt 2.3)
`<plik1>`	pliki tekstowe zawierające kod źródłowy w języku C

Uruchomienie programu bez podania parametrów spowoduje wypisanie powyższych informacji.

2.3 Format pliku z listą ignorowanych funkcji

Jest to plik tekstowy postaci:

```
liczba
ignorowana_funkcja1
ignorowana_funkcja2
ignorowana_funkcja3
...
```

Gdzie:

`liczba`
: dodatnia liczba całkowita informująca i liczbie funkcji zawartych w pliku

`ignorowana_funkcja1, ignorowana_funkcja2, ...`
: kolejne nazwy funkcji do ignorowania rozdzielone białymi znakami

2.4 Wynik działania

Wypisywane jest drzewo wywołań postaci:

```
Plik.c:
     Funkcja foo:
          Wywołuje bar
```

W przypadku wystąpienia błędu praca programu jest przerywana i zostaje wypisany stosowny komunikat.

3. Specyfikacja implementacyjna

3.1 Opis ogólny

Krótki opis modułów:

Moduł	**Pliki**	**Opis**
main	`main.c`	Sterowanie programem
struktury	`struktury.h` `struktury.c`	Struktury używane do przechowywania danych i funkcje do ich obsługi
parser	`parser.h parser.c`	Funkcja parsująca plik
wydruk	`wydruk.h wydruk.c`	Funkcja wypisująca drzewo wywołań

Program korzysta z następujących bibliotek systemowych:
stdio.h
stdlib.h
string.h
ctype.h
limits.h

3.2 Main

Zadaniem **main** jest parsowanie argumentów z jakimi został uruchomiony program, a następnie wywołanie funkcji generującej drzewo z modułu **parser** i wypisującej drzewo z modułu **wydruk** dla każdego podanego pliku.

Moduł nie udostępnia żadnych funkcji.

3.3 Struktury

Zawiera definicje następujących struktur:

```
/*****************************************************
     struktura przechowująca plik do parsowania
*****************************************************/

typedef struct {
     char *nazwa;  //nazwa pliku
     long rozmiar; //rozmiar w charach,
                   //bez końcowego terminatora
     char *kod;    //treść pliku
} kodplik_t;

/*****************************************************
     struktura przechowująca ignorowane funkcje
*****************************************************/

typedef struct {
     unsigned int liczba_funkcji;
     char **lista_funkcji;
} ignfunkcje_t;
```

```
/*****************************************************
     struktura do tworzenia drzewa wywołań
*****************************************************/

//typ węzła drzewa
typedef enum {
     D_PLIK,         //plik
     D_FUNKCJA,      //funkcja i jej treść
     D_FUNKCJA_WYW, //wywołanie funkcji
     D_MAKRO         //makro
} typdrzewa_t;

typedef struct drzewo_t {
     typdrzewa_t typ;
     char *nazwa;
     long zakres[2]; //wektor z numerami znaków:
                     //początkowego i końcowego
     unsigned int liczba_potomkow;
     struct drzewo_t **potomki; //tablica wskaźników
                                //do potomków
} drzewo_t;
```

Udostępnia następujące funkcje służące do ich obsługi:

```
//wczytanie pliku do struktury (konstruktor)
kodplik_t *
kodplik_load(const char *nazwa_pliku);

//destruktor
void
kodplik_free(kodplik_t *dane);

//wczytanie listy ignorowanych funkcji do struktur
//(konstruktor)
ignfunkcje_t *
ignfunkcje_load(const char *nazwa_pliku);

//destruktor
void
ignfunkcje_free(ignfunkcje_t *dane);
```

```
//konstruktor pojedynczego elementu drzewa
drzewo_t *
drzewo_create(const typdrzewa_t typ, const char *nazwa, const
long poczatek, const long koniec);

//destruktor
void
drzewo_free(drzewo_t *drzewo);

//połączenie dwóch drzew
drzewo_t *
drzewo_add(drzewo_t *rodzic, drzewo_t *potomek);
```

3.4 Parser

Udostępnia funkcję generującą drzewo wywołań z podanego pliku:

```
drzewo_t *
generuj_drzewo(kodplik_t *plik, const ignfunkcje_t
*ignorowane_funkcje, const char ignoruj_makra);
```

Oprócz tego zawiera następujące struktury i funkcje prywatne:

```
/**************************
          Struktury
**************************/

//Lista możliwych stanów podczas parsowania - patrz
//funkcja 'preparsuj'
typedef enum {
     ST_KOD, ST_MAKRO, ST_KOMENTARZ_C, ST_KOMENTARZ_CPP,
ST_STRING
} stan_t;

//Lista makr
typedef struct {
     unsigned int liczba;
     char **nazwy;
} listamakr_t;
```

```
//Opis pojedynczej funkcji / wywołania funkcji /
//wywołania makra
typedef struct {
     char *nazwa;
     typdrzewa_t typ;
     unsigned long znak_start;  //początek od znaku
     unsigned long znak_stop;   //koniec na znaku
     unsigned long linia_start; //początek w linii
     unsigned long linia_stop;  //koniec w linii
} funkcja_t;

//Lista funkcji
typedef struct {
     unsigned int liczba;
     funkcja_t **funkcje;
} listafunkcji_t;

/**************************
     Obsługa listamakr_t
**************************/

//konstruktor
listamakr_t *
listamakr_create(void);

//destruktor
void
listamakr_free(listamakr_t *makra);

//dodaje nowe makro "nazwa" do listy makr "makra"
listamakr_t *
listamakr_add(listamakr_t *makra, const char *nazwa);

//drukuje listę makr
void
listamakr_print(FILE *wyj, const listamakr_t *makra);
```

```
/**************************
     Obsługa funkcja_t
**************************/

//konstruktor
funkcja_t *
funkcja_create(const char *nazwa, const typdrzewa_t typ, const
unsigned long znak_start, const unsigned long znak_stop,
     const unsigned long linia_start, const unsigned long
linia_stop);

//destruktor
void
funkcja_free(funkcja_t *funkcja)

/**************************
     Obsługa listafunkcji_t
**************************/

//konstruktor
listafunkcji_t *
listafunkcji_create(void);

//destruktor
void
listafunkcji_free(listafunkcji_t *listafunkcji);

//podpina potomka do rodzica
listafunkcji_t *
listafunkcji_add(listafunkcji_t *rodzic, const funkcja_t
*potomek);

/*****************************************
               Pozostałe funkcje
*****************************************/

//zlicza liczbę linii pomiędzy wskazanymi miejscami w pliku
unsigned long
zlicz_linie(const kodplik_t *plik, const unsigned long
znak_start, const unsigned long znak_stop);

//parsuje nazwę makra
//"start" to wskaźnik do pierwszego znaku za '#'
//"ile" to liczba znaków z ilu składa się dana do
//parsowania dyrektywa
char *
parsuj_nazwe_makra(char *start, unsigned int ile);
```

```
//wyłapuje i kasuje makra oraz usuwa komentarze i stringi
//zwraca 0 gdy wszystko OK
int
preparsuj(kodplik_t *plik, listamakr_t *makra);

//parsuje i zwraca nazwę funkcji, zwraca NULL
//w przypadku niepowodzenia/gdy to nie jest funkcja
char *
to_funkcja(kodplik_t *plik, unsigned long znak);

//wykrywa funkcje w pliku
listafunkcji_t *
wykryj_funkcje(kodplik_t *plik);

//sprawdza czy dana funkcja jest na podanej
//liście ignorowanych
char
czy_ignorowana(const ignfunkcje_t *ignorowane, const char
*nazwafunkcji);

//sprawdza czy dane wywołanie funkcji jest makrem
typdrzewa_t
to_makro(const listamakr_t *makra, const char *nazwafunkcji);

//parsuje i zwraca nazwę wywoływanej funkcji,
//zwraca NULL w przypadku
//niepowodzenia/gdy to nie jest wywołanie funkcji
char *
to_wywolanie_funkcji(kodplik_t *plik, const funkcja_t *funkcja,
unsigned long znak);

//rekurencyjnie wykrywa wywołania funkcji w danej funkcji
//opcjonalnie uwzględnia ignorowane i informowanie o makrach
drzewo_t *
wykryj_wywolania(kodplik_t *plik, const funkcja_t *funkcja,
const ignfunkcje_t *ignorowane, const listamakr_t *makra);
```

3.5 Wydruk

Udostępnia funkcję wypisującą drzewo wywołań do podanego pliku:

```
void
drukuj_drzewo(FILE *wyjscie, const drzewo_t *drzewo, const char
wypisuj_numery);
```

Oprócz tego zawiera następujące funkcje prywatne:

```
//zwraca tekstowy opis typu węzła drzewa
char *
typdrzewatostr(typdrzewa_t typ);

//zwraca string złożony z "liczba" znaków "znak"
char *
strofchar(const char znak, const unsigned int liczba);

//rekurencyjnie drukuje drzewo
void
drukuj_drzewo_r(FILE *wyjscie, const drzewo_t *drzewo, const
char wypisuj_numery, unsigned int glebokosc);
```

4. Testy

4.1 Kompilacja

System operacyjny: OpenSUSE 11.3 z jądrem w wersji 2.6.34
Kompilator: gcc 4.5

4.2 Działanie

Działanie programu było testowane przy wykorzystaniu jego plików źródłowych (w tym również celowo zmodyfikowanych, by zbadać reakcję na błędy) oraz losowo wybranych modułów z jądra Linux w wersji 2.6.37

Nie stwierdzono żadnych błędów w działaniu.