

Zad 1.

piątek, 5 maja 2023 13:15

Zadanie 1. Poniżej podano zawartość pliku «swap.c». Wskaż w nim wszystkie wystąpienia definicji i referencji do **symboli** [1, §7.5]. Dla każdego symbolu wskaż jego **zasięg widoczności** (tj. lokalny, globalny, zewnętrzny) oraz nazwę **sekcji**, w której go umieszczono (tj. «.text», «.data», «.rodata», «.bss»). Wydając polecenie «make swap.o» wygeneruj **plik relokowalny** i zweryfikuj swoje odpowiedzi na podstawie wydruku z polecenia `nm`¹. Do czego *konsolidator* wykorzystuje tablicę symboli?

```

1 extern int printf(      text 7 static void incr() {      text 16 void swap(int i) {
2   const char *, ...);  bss 8   static int count = 0;      17   incr();
3 extern long buf[];      9   count++;      18   long temp = *bufp0;
4                               10 }      19   *bufp0 = buf[i];
5 long *bufp0 = &buf[0];  11 void addf(void) {      20   buf[i] = temp;
6 static double sum = 0.0; 12   sum += 3.14;      21 }
                          13   printf("sum = %f\n", sum);
                          14 }
                          15 }

```

Handwritten annotations:

- `1 extern int printf(...);`: undef (green dot)
- `3 extern long buf[];`: undef (green dot)
- `5 long *bufp0 = &buf[0];`: data (orange dot)
- `6 static double sum = 0.0;`: bss (pink dot)
- `7 static void incr() {`: text (pink dot)
- `8 static int count = 0;`: bss (pink dot)
- `9 count++;`: text (pink dot)
- `10 }`: text (pink dot)
- `11 void addf(void) {`: text (pink dot)
- `12 sum += 3.14;`: text (pink dot)
- `13 printf("sum = %f\n", sum);`: text (pink dot)
- `14 }`: text (pink dot)
- `16 void swap(int i) {`: text (orange dot)
- `17 incr();`: text (orange dot)
- `18 long temp = *bufp0;`: text (orange dot)
- `19 *bufp0 = buf[i];`: text (orange dot)
- `20 buf[i] = temp;`: text (orange dot)
- `21 }`: text (orange dot)

symbole

- █ - definicja
- █ - referencja

zasięg

- - lokalny
- - globalny
- - zewnętrzny

```

mluczynski@mluczynski:~/Desktop/studia/ask/Lista 8/lista_8$ nm swap.o
0000000000000000 U buf
0000000000000000 D bufp0
0000000000000000 b count.0
0000000000000000 U _GLOBAL_OFFSET_TABLE_
0000000000000000 t incr
0000000000000000 r .LC0
0000000000000000 r .LC1
0000000000000000 U printf
0000000000000008 b sum
0000000000000039 T swap

```

Handwritten notes:

- to jest w bss, bo = 0.0 (pointing to `b sum`)
- ~ T addf (pointing to the missing `addf` symbol)
- ← tego nie ma w swap.c (pointing to the missing `addf` symbol)

An **object file** will contain a symbol table of the identifiers it contains that are externally visible. During the linking of different object files, a linker will identify and resolve these symbol references. Usually all undefined external symbols will be searched for in one or more object libraries. If a module is found that defines that symbol it is linked with together with the first object file, and any undefined external identifiers are added to the list of identifiers to be looked up. This process continues until all external references have been resolved. It is an error if one or more remains unresolved at the end of the process.

do tego jest tabela symboli ↗

.bss *Uninitialized* global and static C variables, along with any global or static variables that are initialized to zero. This section occupies no actual space in the object file; it is merely a placeholder. Object file formats distinguish between initialized and uninitialized variables for space efficiency: uninitialized variables do not have to occupy any actual disk space in the object file. At run time, these variables are allocated in memory with an initial value of zero.