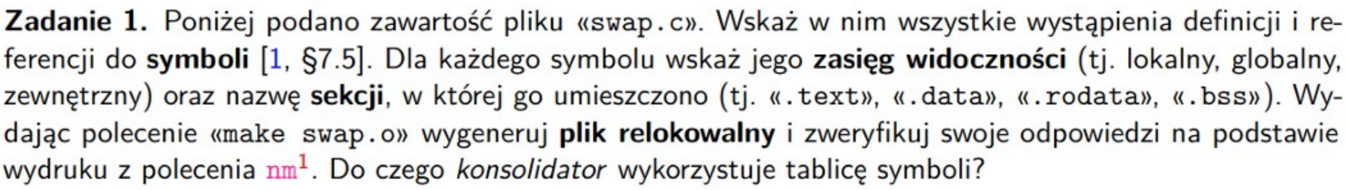
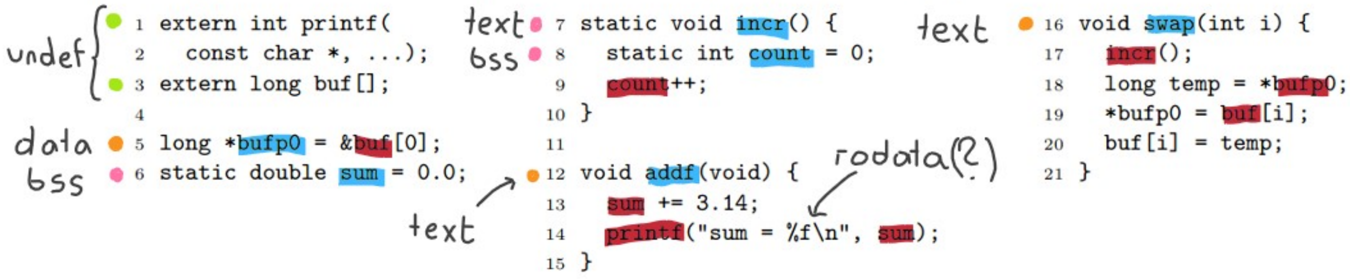
Zad 1.

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?



```
extern int printf(
  const char *, ...);
extern long buf[];

long *bufp0 = &buf[0];
static double sum = 0.0;

static void incr() {
  static int count = 0;
  count++;
}

void addf(void) {
  sum += 3.14;
  printf("sum = %f\n", sum);
}

void swap(int i) {
  incr();
  long temp = *bufp0;
  *bufp0 = buf[i];
  buf[i] = temp;
}
```

symbole

🟦 – definicja

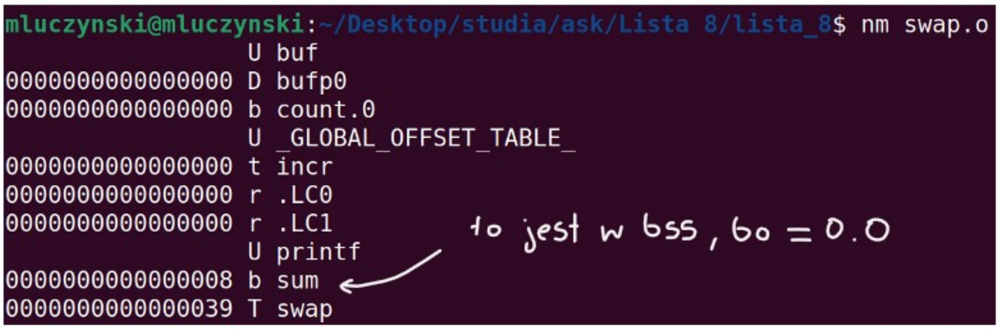
🟥 – referencja

zasięg

● – lokalny

🟠 – globalny

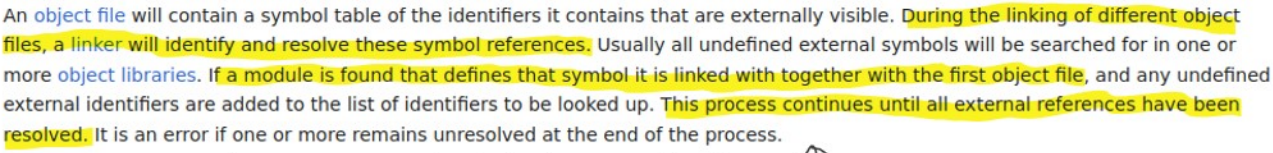
🟢 – zewnętrzny



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

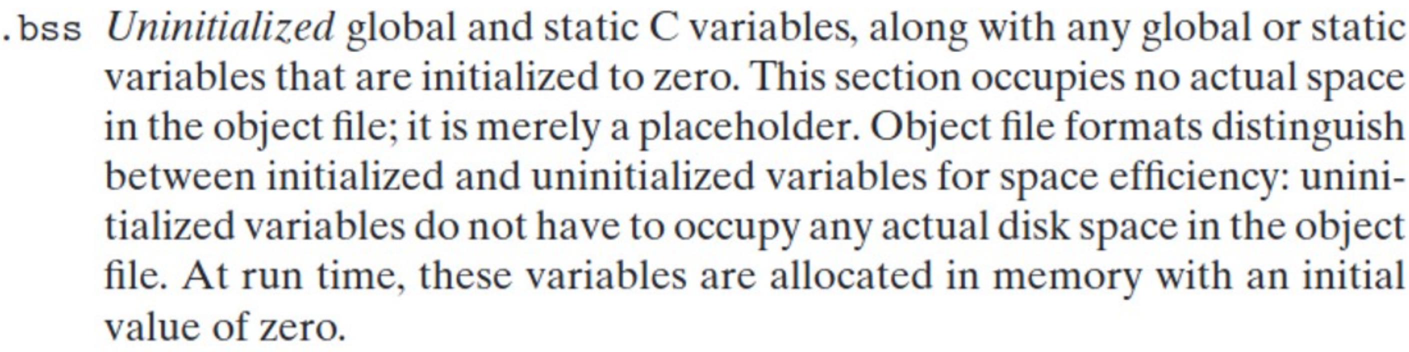
to jest w bss, bo = 0.0

~ T addf ↖ tego nie ma w swap.c



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.