WOJSKOWA AKADEMIA TECHNICZNA
im. Jarosława Dąbrowskiego
Wydział Elektroniki, Instytut Telekomunikacji
ul. Gen. Sylwestra Kaliskiego 2, 00-908 Warszawa

PRZEDMIOT „PROGRAMOWANIE W C/C++ DLA ZASTOSOWAŃ SIECIOWYCH” STACJONARNE STUDIA I°

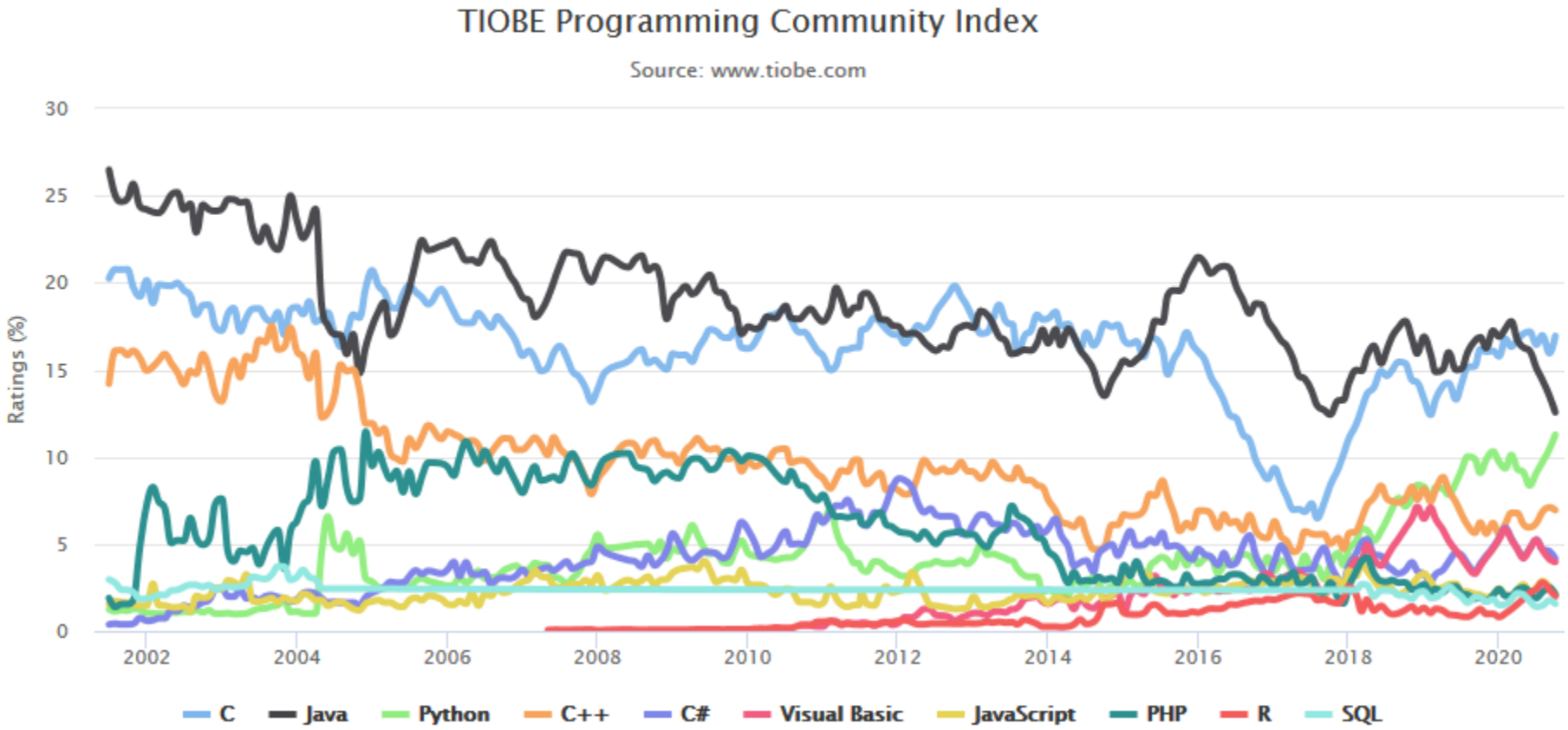
T: 1

dr inż. Jarosław KRYGIER
p. 122 b.47, tel. 22 6837193
email: jaroslaw.krygier@wat.edu.pl

Materiały dydaktyczne: jkrygier.wel.wat.edu.pl

Popularność języka C/C++ (do 10.2020)

https://www.tiobe.com/tiobe-index/



C

1970r. - Dennis Ritchie (Bell Laboratories)
1989r. – Zestandaryzowany przez ANSI (American National Standards Institute) – ANSI C (C89)
1990r. – ratyfikowany przez ISO standard ANSI C – C90
1995r. – aktualizacja standardu ISO – C95
1999r. – aktualizacja standardu ISO – C99
2011r. – aktualizacja standardu ISO – C11

Aktualna wersja:
2018r. – C18 (ISO/IEC 9899:2018)
https://www.iso.org/obp/ui/#iso:std:iso-iec:9899:ed-4:v1:en

Draft standardu C18 (dostępny bezpłatnie):
https://web.archive.org/web/20181230041359if_/http://www.open-std.org/jtc1/sc22/wg14/www/abq/c17_updated_proposed_fdis.pdf

Draft najnowszego standardu C2x:
http://www.open-std.org/jtc1/sc22/wg14/www/docs/n2573.pdf

C++

1998r. - ISO/IEC 14882:1998 – C++98
2003r. - ISO/IEC 14882:2003 - C++03
2011r. - ISO/IEC 14882:2011 - C++11
2014r. - ISO/IEC 14882:2014 - C++14

Aktualna wersja:
2017r. – C++17 (ISO/IEC 14882:2017)
https://www.iso.org/standard/68564.html

Dostępna bezpłatna wersja C++17 (przed oficjalną płatną):
http://www.open-std.org/jtc1/sc22/wg21/docs/papers/2017/n4713.pdf

Wojskowa Akademia Techniczna im. Jarosława Dąbrowskiego

Środowisko programistyczne

- Biblioteki systemowe i nagłówkowe (np. `glibc`, `/usr/include`)
- Źródła programu (pliki .c .h)
- Kompilator (.o, .obj)
- Linker (pliki wykonywalne)

Struktura programu

Musi być funkcja główna
Przykłady:

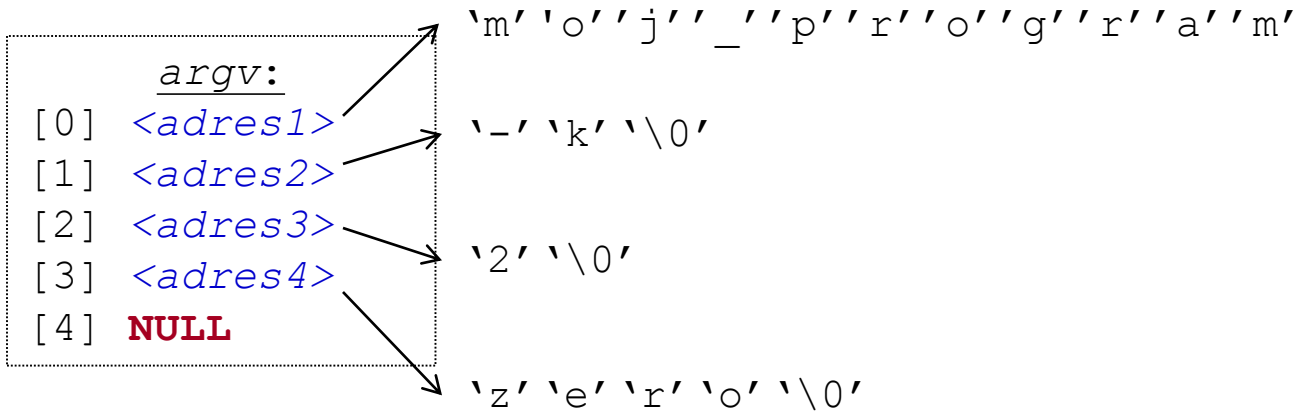
```
int main (void)
{
...
...
}

int main (int argc, char *argv[]) {
...
}
```

Struktura programu – argumenty funkcji głównej

```
int main (int argc, char *argv[]) {…}
int main (int argc, char **argv) {…}
```

```
./moj_program -k 2 zero
```

argc = 4,
argv = *<adres 0 komórki>*



Przykład

```
#include <stdio.h>
#include <stdlib.h>

int main (void)
{
    printf("Hello World\n");
    exit(0);
}
```

Tutaj jest zdefiniowana

```
void exit(int status);
```

przerywa działanie programu opróżnia wszystkie bufory i zwraca kod `status` do systemu operacyjnego

Wojskowa Akademia Techniczna im. Jarosława Dąbrowskiego

stdlib.h - przykłady funkcji

```
void *malloc(size_t size);
void *realloc(void *ptr, size_t size);
void free (void *ptr);
void exit(int status);
int system(const char *string);
int rand(void);
double atof(const char *str);
int atoi(const char *str);
long int atol(const char *str);
```

Dyrektywy preprocesora

Podstawowe
#include
#define
#undef
Warunkowe
#if
#elif
#else
#endif
#ifdef
#ifndef

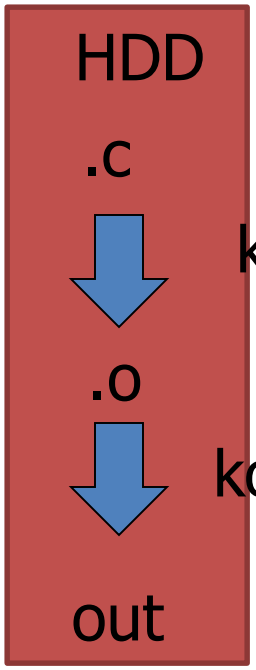
```
/*przykład 1*/
#define SUMA (a,b) a+b
#define STALA 2
#define SLOWO_KLUCZOWE
#include <plik_nagłówkowy.h>
```

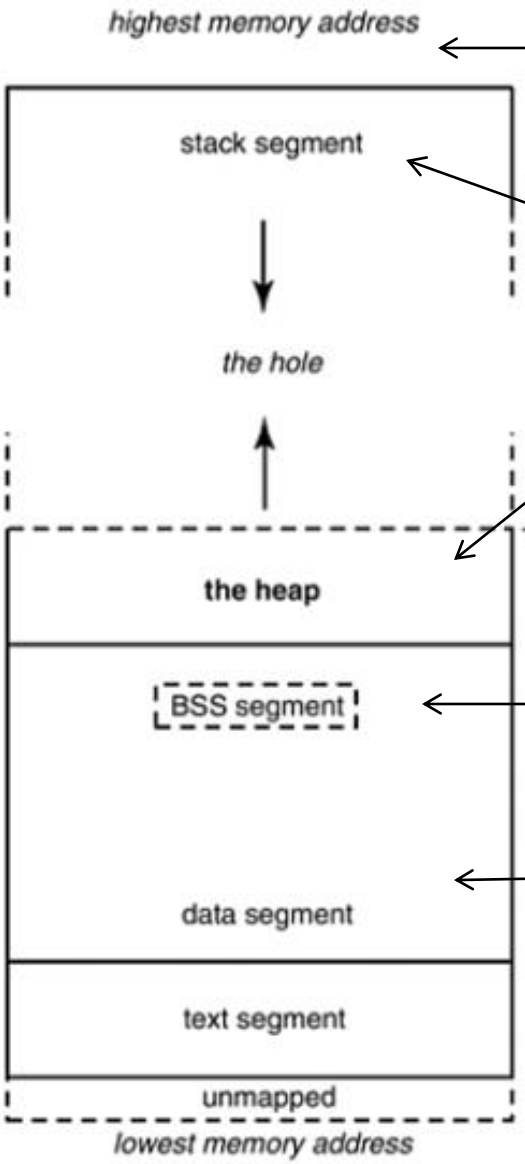
/usr/include/stdio.h

```
/* comments */
#ifndef _STDIO_H
#define _STDIO_H

... definitions and protoypes

#endif
```

```
/*przyklad2*/
#if STALA == 2
/*wykonaj kod 2*/
#elif STALA == 1
/*wykonaj kod 1*/
#else
/*wykonaj kod 0*/
#endif
/*wykonaj reszte*/
```

Usadowienie programu w pamięci





Zmienne linii komend

Lokalne zmienne

Dynamicznie alokowana pamięć

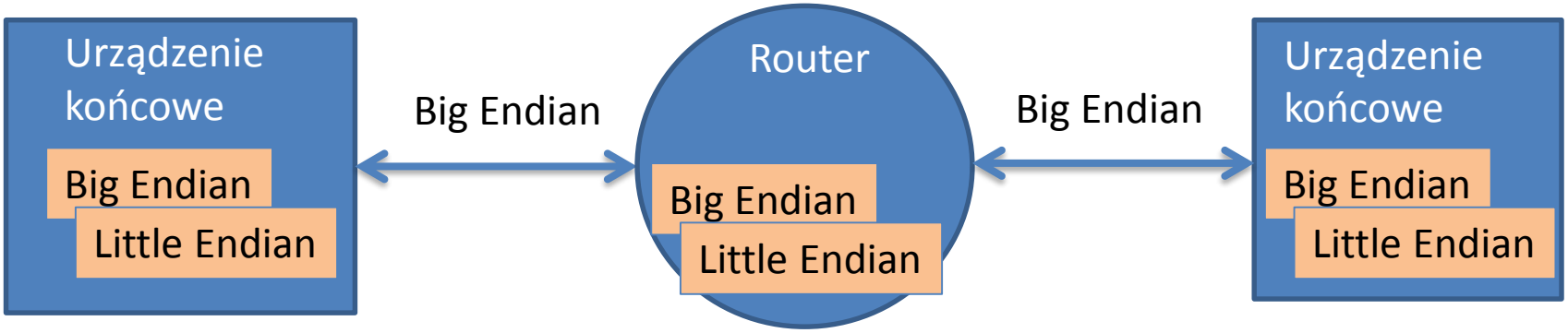
used for `malloc()`

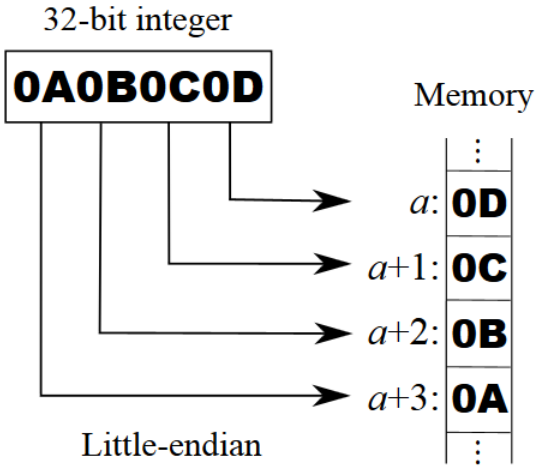
Dane programu niezainicjowane (BSS - Block Storage Start) - (global, static, constant, external)

Dane programu zainicjowane (global, static, constant, external)

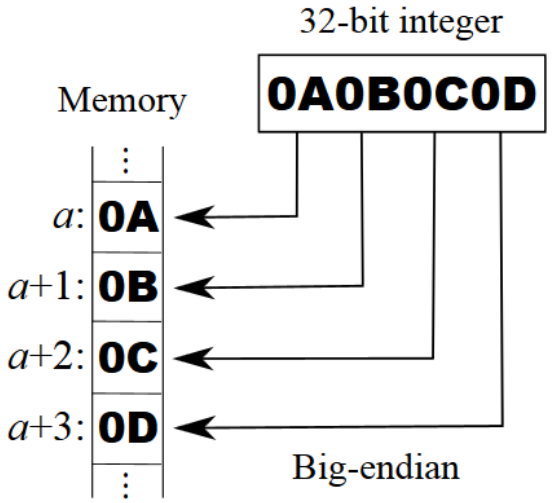
Kod programu

Format przesyłania/zapisu danych



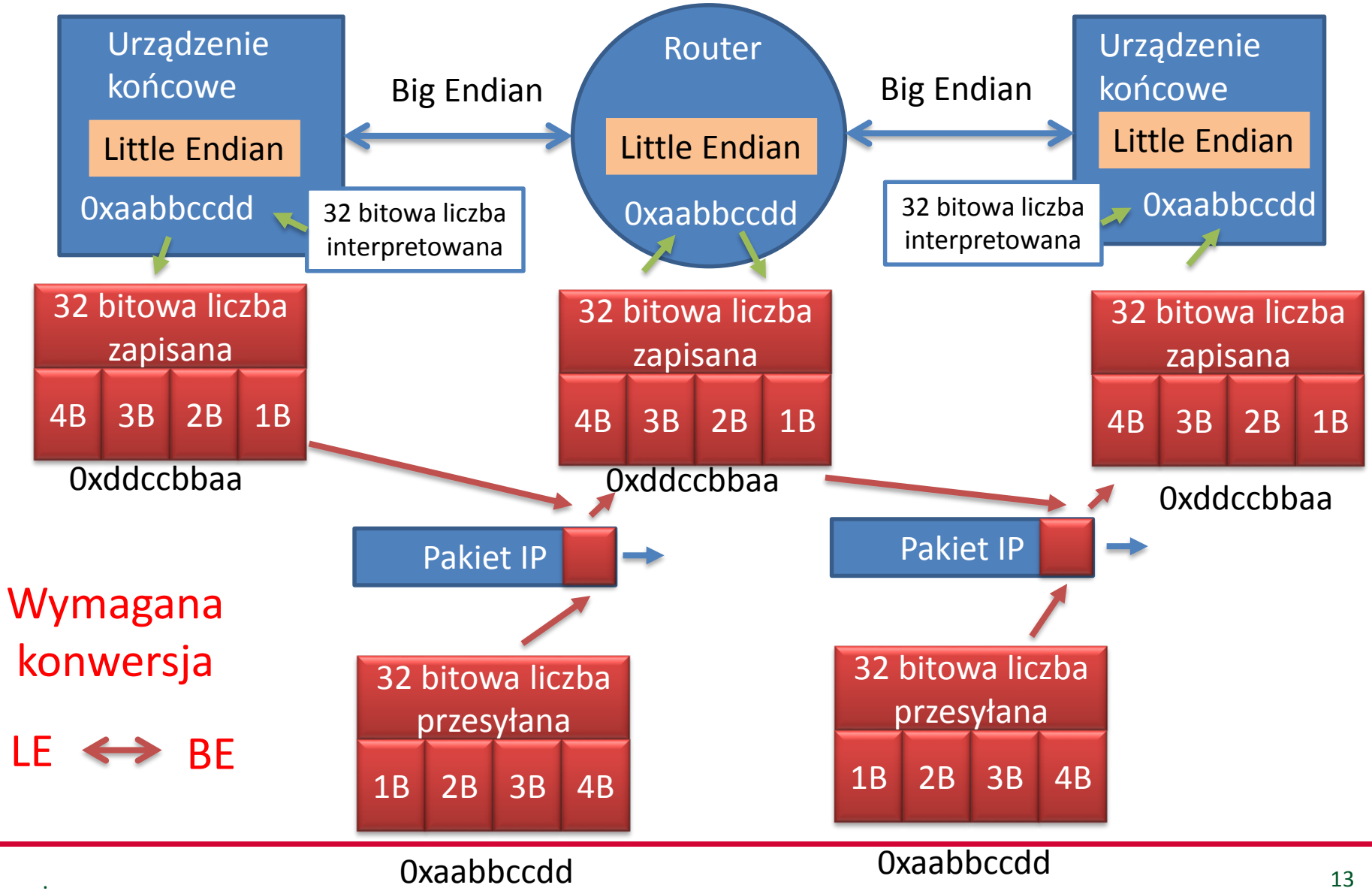


https://en.wikipedia.org/wiki/Endianness



https://en.wikipedia.org/wiki/Endianness

Format przesyłania/zapisu danych



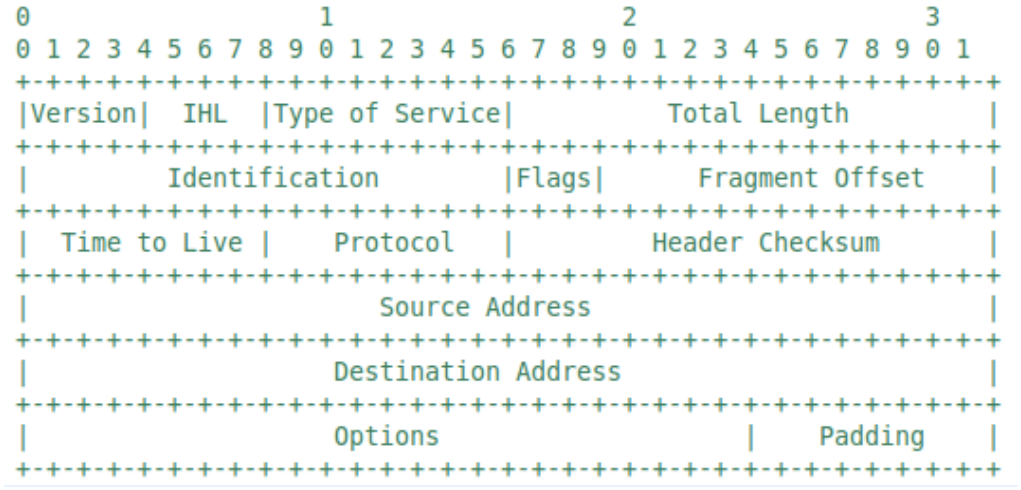
Podstawowe typy danych wykorzystywane w aplikacjach sieciowych (SO 32b)

char	8 bitów (1B)
short int	16 bitów (2B)
int	32 bity (4B)
long int	32 bity (4B)
long long int	64 bity (8B)

unsigned char	8 bitów (1B)
unsigned short int	16 bitów (2B)
unsigned int	32bity (4B)
unsigned long int	32 bity (4B)
unsigned long long int	64 bity (8B)

float	32 bity (4B)
double	64 bity (8B)
long double	96 bity (12B)

Formaty zapisu nagłówków



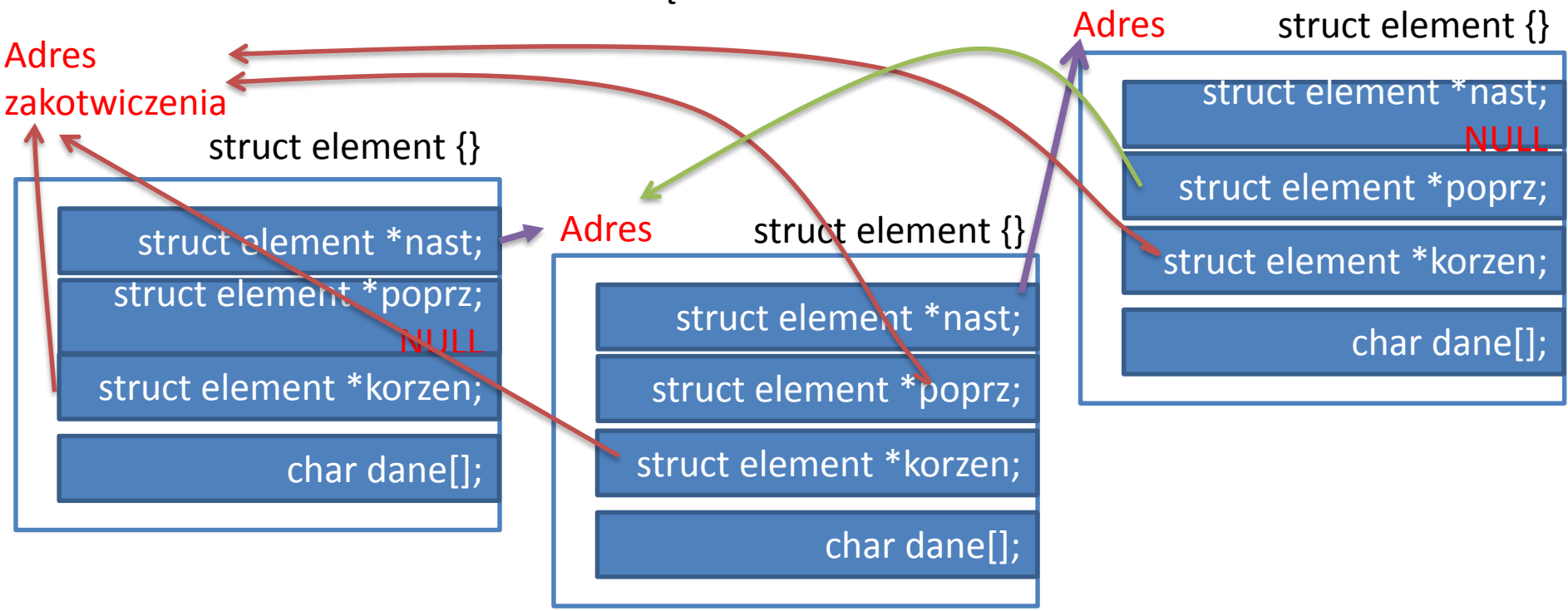

```
struct iphdr
 {
#if __BYTE_ORDER == __LITTLE_ENDIAN
   unsigned int ihl:4;
   unsigned int version:4;
#elif __BYTE_ORDER == __BIG_ENDIAN
   unsigned int version:4;
   unsigned int ihl:4;
#endif
   u_int8_t tos;
   u_int16_t tot_len;
   u_int16_t id;
   u_int16_t frag_off;
   u_int8_t ttl;
   u_int8_t protocol;
   u_int16_t check;
   u_int32_t saddr;
   u_int32_t daddr;
};
```

Bufory

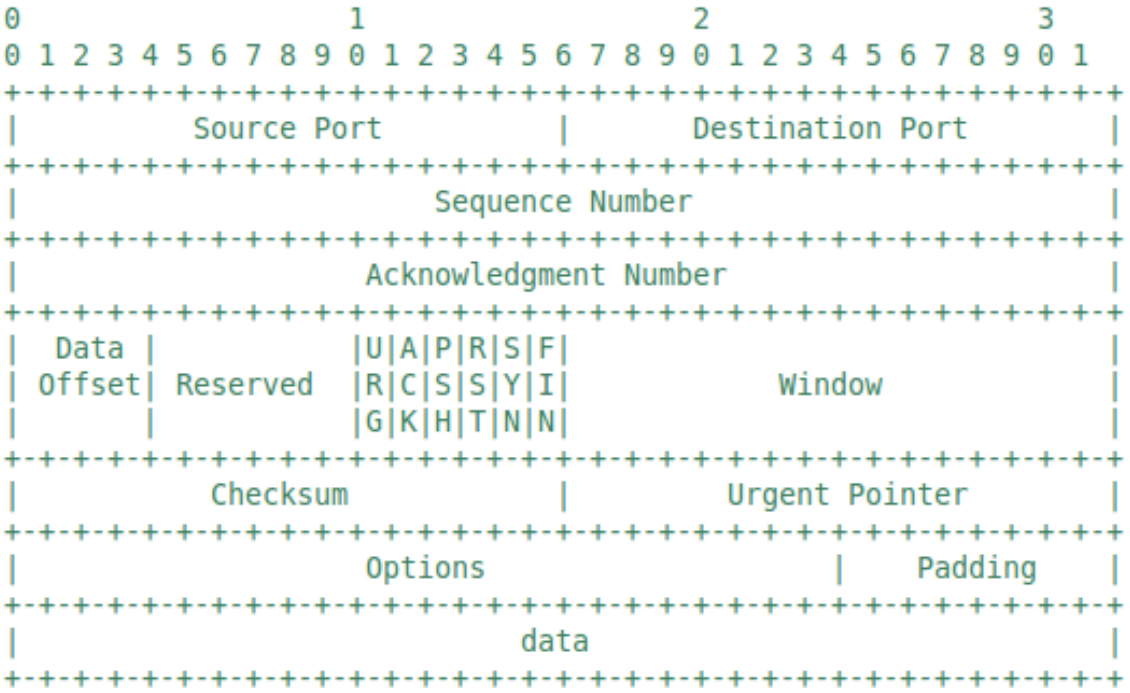
Bufor znakowy

```
#define ROZMIAR BUFORA 1518;

unsigned char        bufor_znakowy [ROZMIAR BUFORA];
```

Lista wiązana



Flagi



Nagłówek TCP

Uwaga: Little Endian

```
struct tcphdr
 {
   u_int16_t source;
   u_int16_t dest;
   u_int32_t seq;
   u_int32_t ack_seq;
   u_int16_t res1:4;
   u_int16_t doff:4;
   u_int16_t fin:1;
   u_int16_t syn:1;
   u_int16_t rst:1;
   u_int16_t psh:1;
   u_int16_t ack:1;
   u_int16_t urg:1;
   u_int16_t res2:2;
   u_int16_t window;
   u_int16_t check;
   u_int16_t urg_ptr;
};
```

Wskaźniki na funkcję

Deklaracja wskaźnika na funkcję

int (*funkcja) (int, int);

Można:

- przekazywać do innych funkcji jako argumenty
- zwracać jako wynik działania funkcji
- porównywać z NULL

Nie można

- wykonywać operacji matematycznych

Biblioteki dynamiczne

//Utworzenie biblioteki dynamicznej:
//gcc -shared biblioteka.c -o biblioteka.so

Aby mieć 'dynamic library' (dl), linker musi łączyć z opcja -l: dl
Ustawić w Eclipsie: C/C++ -> Settings ->GCC C Linker -> Libraries
Program może ładować w trakcie pracy potrzebne mu biblioteki. Linker dostarcza w tym celu 4 funkcji:

- dlopen ładowanie biblioteki,
- dlsym zwrócenie wskaźnika do odpowiedniego symbolu (funkcji) w bibliotece,
- dlerror obsługa błędów,
- dlclose zamykanie biblioteki.

Funkcje te są udostępniane poprzez nagłówek dlfcn.h (trzeba go dodać do głównego programu).

```
#include <dlfcn.h>
void *dlopen(const char *filename, int flag);
char *dlerror(void);
void *dlsym(void *handle, const char *symbol);
int dlclose(void *handle);
```