Programozás C nyelven (10. ELŐADÁS)

Sapientia EMTE

2020-21

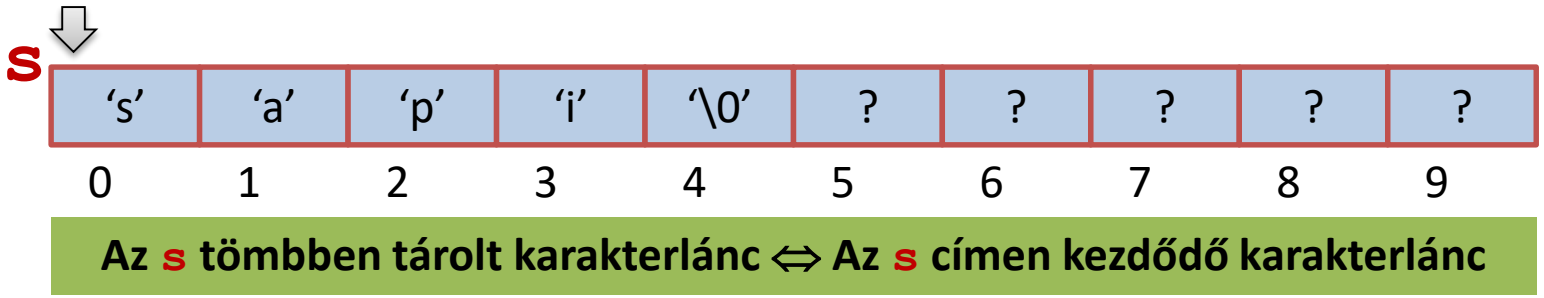
Olvasás/írás szöveges állományból/ba

```
FILE *fin, *fout;
fin = fopen("szoveg.txt", "r");
if( fin == NULL ){
   printf("Sikertelen input allomany megnyitas!");
   return 0;
}
… fscanf(fin, "…", …); …
fclose(fin);

fout = fopen("eredmeny.txt", "w"); if(…){…}
… fprintf(fout, "…", …); …
fclose(fout);
```

Karakterláncok

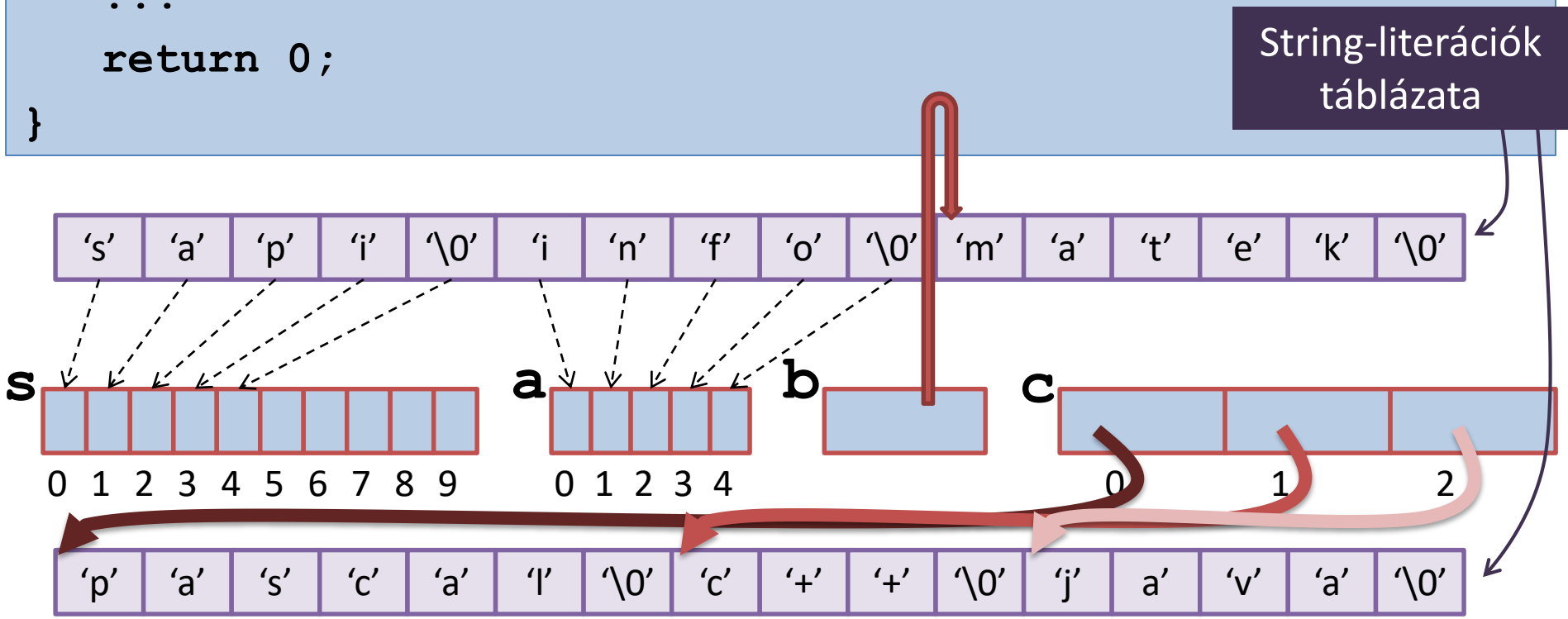
- A karakterláncokat `char`-tömbökben tároljuk.
- Egy karakterlánc végét a karakterlánc-végjel jelzi: `'\0'`

```
int main(){
    char s[10] = "sapi";
    ...
    return 0;
}
```



Karakterláncok

```
int main(){
    char s[10] = "sapi"; //sizeof(s)=10
    char a[] = "info"; //sizeof(a)=5
    char *b = "matek"; //sizeof(b)=4
    char *c[] = {"pascal", "c++", "java"}; //sizeof(c)=12
    ...
    return 0;
}
```



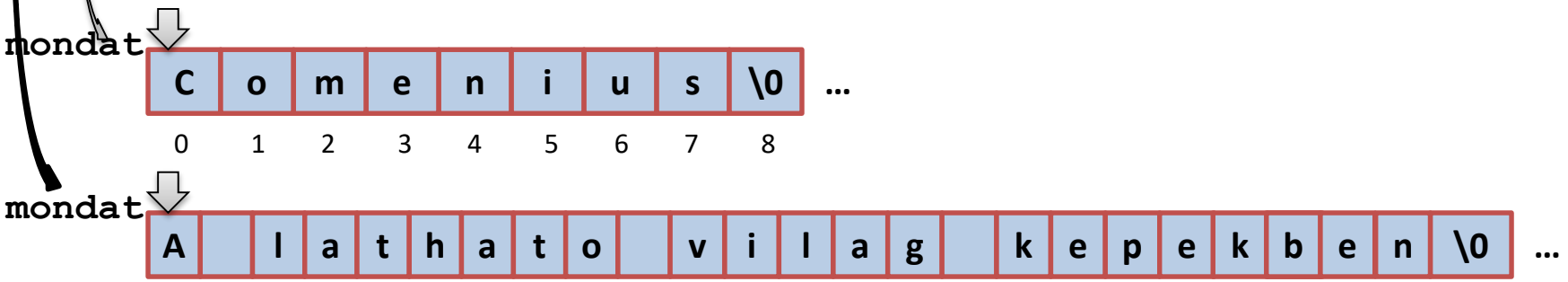
Karakterláncok beolvasása / kiírása

```
int main(){
    char mondat[100];
    scanf("%s", mondat); printf("%s\n", mondat);
    FILE *fin;
    fin = fopen("szoveg.txt", "r"); if(…)
    fscanf(fin, "%[^\n]\n", mondat);
    printf("%s\n", mondat);
    fscanf(fin, "%[^\n]\n", mondat);
    printf("%s\n", mondat);
    fclose(fin);
    return 0;
}
```

szoveg.txt

```
Comenius egyik fo muve:
A lathato vilag kepekben
```

```
Comenius a modern oktatas atyja
Comenius
Comenius egyik fo muve:
A lathato vilag kepekben
_
```

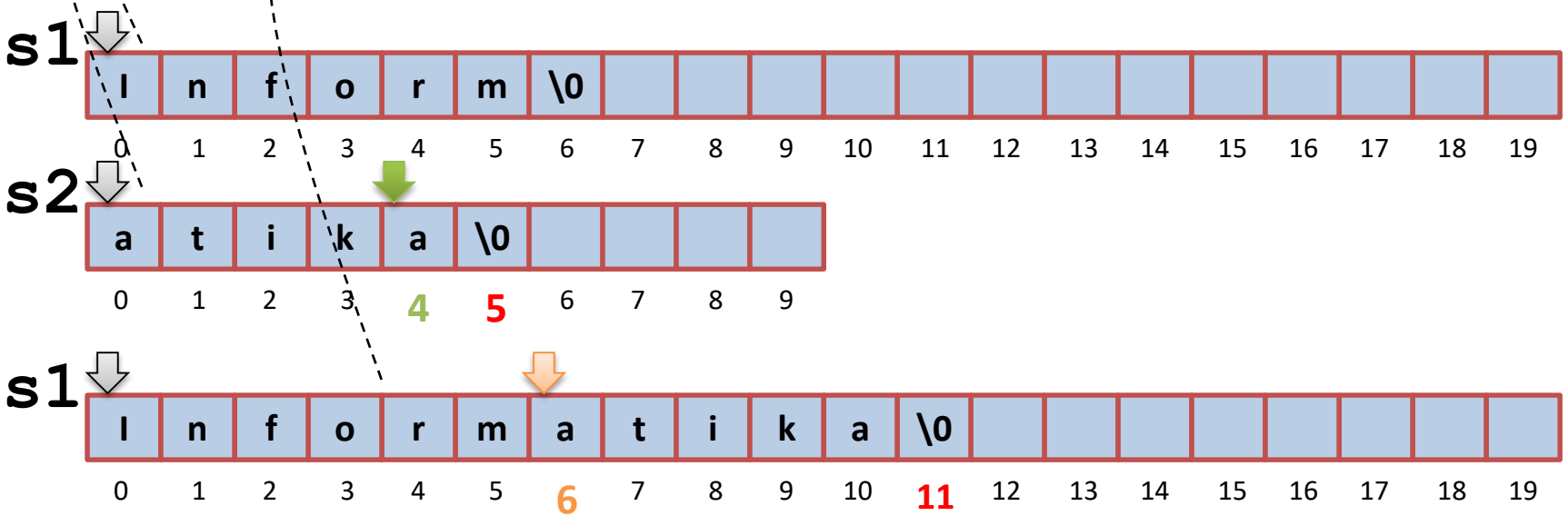


string.h(1)

- `size_t strlen(const char *str);`
  //karakterlánc hossz
- `char *strcpy(char *dest, const char *src);`
  //másolás
- `char *strcat(char *dest, const char *src);`
  //egymásutánfűzés
- `int strcmp(const char *str1, const char *str2);`
  //lexikográfikus összehasonlítás
- `char *strchr(const char *str, int c);`
  //c első előfordulásának címe str-ben
- `char *strrchr(const char *str, int c);`
  // c utolsó előfordulásának címe str-ben
- `char *strstr(const char *str1, const char *str2);`
  //str2 első előfordulásának címe str1-ben

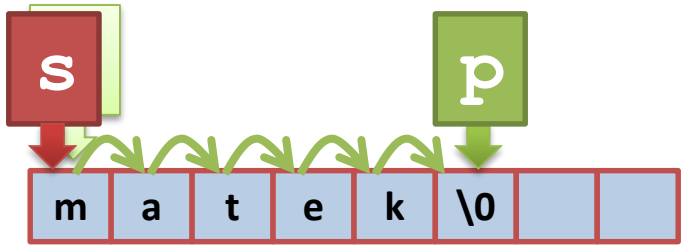
```
#include <string.h>
int main(){
    char s1[20], s2[10];
    strcpy(s1,"Inform");
    strcpy(s2,"atika");
    printf("%s\n", strcat(s1,s2));
    printf("%i,%i\n", strlen(s1), strlen(s2));
    printf("%i\n", strstr(s1,s2)-s1);
    printf("%i\n", strrchr(s2,'a')-s2);
    return 0;
}
```

```
Informatika
11,5
6
4
_
```



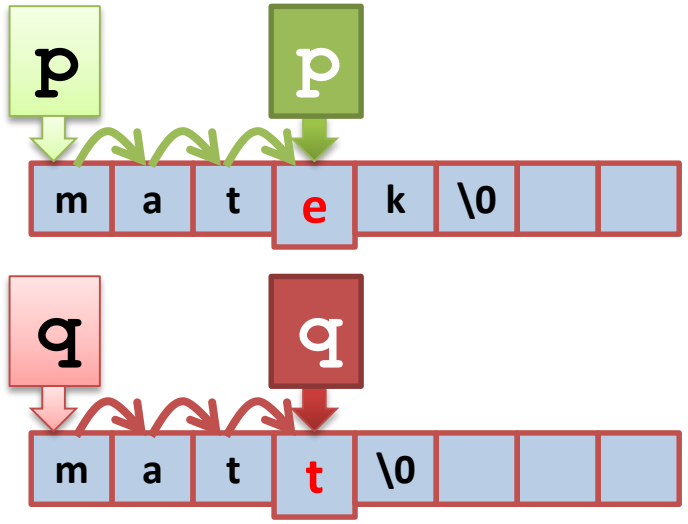
my_strlen / my_strcmp

```
int my_strlen(char *s){
    char *p;
    for( p = s ; *p ; ++p ){;}
    return p - s;
}
```




```
int my_strcmp(char *p, char *q){
    while( *p == *q ){++p; ++q;}
    return *p - *q;
}
```

Milyen esetben nem működik helyesen?



string.h (2)

- `char *strncat(char *dest, const char *src, size_t n);`
  //legfentebb n karakter hozzáfűzése
- `int strncmp(const char*str1, const char*str2, size_t n);`
  // legfentebb n karakter összehasonlítása
- `char *strncpy(char *dest, const char *src, size_t n);`
  // legfentebb n karakter másolása
- `size_t strcspn(const char *str1, const char *str2);`
  //str1 ”str2-mentes” prefixének hossza
- `char *strpbrk(const char *str1, const char *str2);`
  // str1 első ”str2-beli” karaktere címe
- `size_t strspn(const char *str1, const char *str2);`
  //str1 ”str2-beli” prefixének hossza
- `char *strtok(char *str, const char *delim);`
  //karakterlánc darabolás

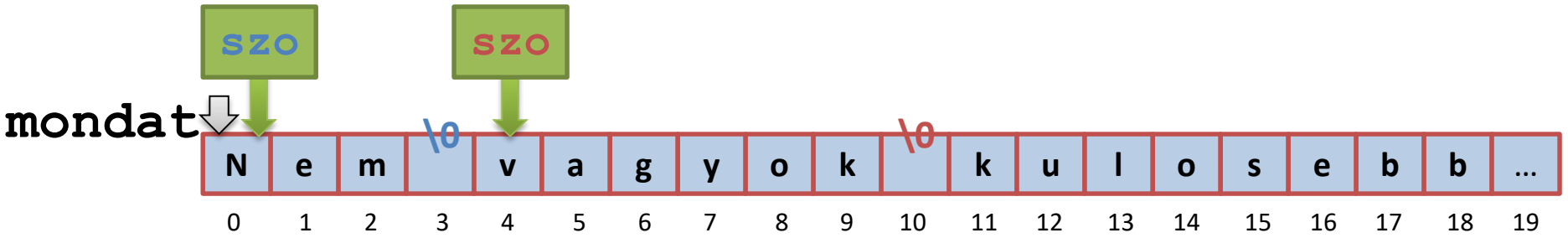
`strtok`: karakterlánc-darabolás

```
char mondat[80] = "Nem vagyok kulonosebben tehetseges,
csak szenvedelyesen kivancsi. (Einstein)";

const char elvalasztok[] = " ,.()";
char *szo;

/* nyerd ki az ELSŐ szót */
szo = strtok(mondat, elvalasztok);

/* nyerd ki a TÖBBI szót */
while( szo != NULL ) {
   printf("%s\n", szo);
   szo = strtok(NULL, elvalasztok);
}
```

Nem
vagyok
kulonosebb
tehetseg
csak
szenvedelyesen
kivancsi
Einstein
_

Az első „szó” kezdőcímét téríti vissza, valamint lezárja ezt ‘\0’-val.

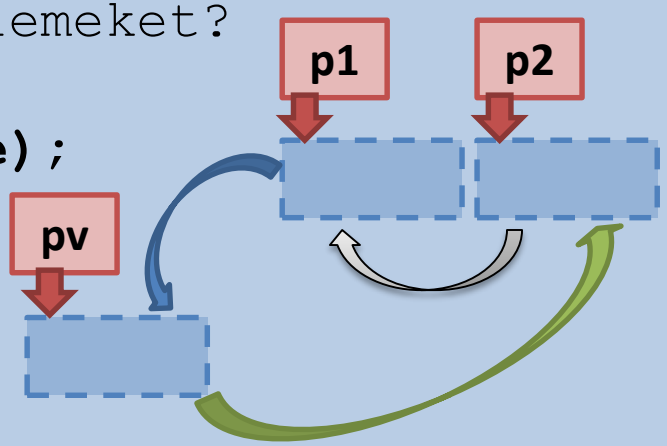
Ha a maradék-karakterlánc üres, vagy csak elválasztó-karaktereket tartalmaz, akkor NULL pointer-t térít vissza.



string.h(3)

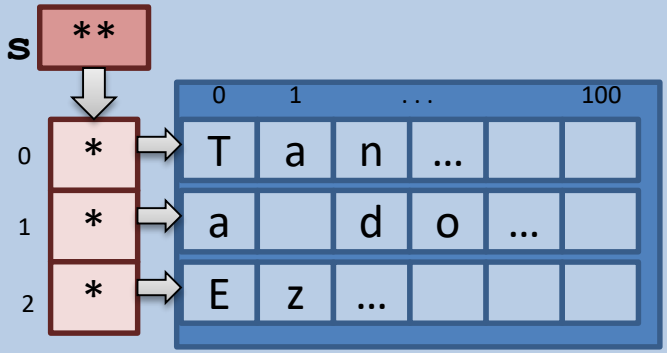
- `void *memchr(const void *str, int c, size_t n);`
- `int memcmp(const void*str1, const void*str2, size_t n);`
- `void *memcpy(void *dest, const void *src, size_t n);`
- `void *memmove(void *dest, const void *src, size_t n);`
- `void *memset(void *str, int c, size_t n);`

```
//Hogyan cseréli ki a qsort a p1 és p2 címeken
//található size méretű elemeket?

void *pv; pv = malloc(size);

memcpy(pv, p1, size);
memcpy(p1, p2, size);
memcpy(p2, pv, size);
```



n darab, maximum 100 karakterű sorok beolvasása állományból kétdimenziós tömbbe

```
#include <stdio.h>
#include <string.h>
int main(){
   int n, i;
   char **s;
   FILE *fin;
   fin = fopen("szoveg.txt", "rt"); if(!fin){…}
   fscanf(fin, "%i\n", &n);
   s = (char**)malloc(n * sizeof(char*)); if(…){…}
   for (i = 0 ; i < n ; ++i){
      s[i] = (char*)malloc(101 * sizeof(char)); if(…){…}
      fscanf(fin, "%[^\n]\n", s[i]);
   }
   fclose(fir
   return 0;
}
```



szoveg.txt

```
3
Tanítani szinte nem is jelent mást, mint megmutatni, miben különböznek egymástól
a dolgok a különböző céljukat, megjelenési formájukat és eredetüket illetően .
Ezért aki jól megkülönbözteti egymástól a dolgokat, az jól is tanít. (Comenius)
```

Szöveget tartalmazó állomány feldolgozása karakterenként

Hány karaktert tartalmaz a „szoveg.txt” állomány?

```
#include <stdio.h>
int main(){
    int i; char c;
    FILE *fin;
    fin = fopen("szoveg.txt", "rt");
    if(!fin){
        printf("Sikertelen allomanymegnyitas");
        return 0;
    }
    i = 0;
    while( fscanf(fin, "%c", &c) != EOF ){
        ++i;
    }
    printf("Karakterek szama: %i", i);
    fclose(fin);
    return 0;
}
```

szoveg.txt

```
Kiss Elemer
Szakkollegium
_
```

Karakterek szama: 26

Sikertelen olvasás esetén az fscanf az EOF konstant téríti vissza

Szöveget tartalmazó állomány feldolgozása szavanként

Hány szót tartalmaz a „szoveg.txt” állomány?

```
#include <stdio.h>
int main(){
    int i; char szo[30];
    FILE *fin;
    fin = fopen("szoveg.txt", "rt");
    if(!fin){
        printf("Sikertelen allomanymegnyitas");
        return 0;
    }
    for( i = 0 ; fscanf(fin, "%s", szo) != EOF ; ++i ){
        ;
    }
    printf("Szavak szama: %i", i);
    fclose(fin);
    return 0;
}
```

szoveg.txt

```
Kiss Elemer
Szakkollegium
_
```

```
Szavak szama: 3
```

Szöveget tartalmazó állomány feldolgozása soronként

Hány sort tartalmaz a „szoveg.txt” állomány?

```
#include <stdio.h>
int main(){
    int i; char sor[100];
    FILE *fin;
    fin = fopen("szoveg.txt", "rt");
    if(!fin){
        printf("Sikertelen allomanymegnyitas");
        return 0;
    }
    for( i = 0 ; fscanf(fin, "%[^\n]\n", sor) != EOF ; ++i ){
        ;
    }
    printf("Sorok szama: %i", i);
    fclose(fin);
    return 0;
}
```

szoveg.txt

```
Kiss Elemer
Szakkollegium
_
```

Sorok szama: 2

Összefoglalás

- A karakterláncokat `char`-tömbökben tároljuk.
- Egy karakterlánc végét a karakterlánc-végjel jelzi: `'\0'`

- C olvasás „szavanként” (`%s`) / soronként (`%[^\n]\n`)

- **`string.h`**
  - `strlen, strcpy, strcat, strchr, strstr, strcmp, …`
  - `strcspn, strpbrk, strspn, strtok, …`
  - `memcpy, memmov, memcmp, …`