Programowanie strukturalne - Wykład 7

Tablice jednowymiarowe

Czy można zwrócić tablicę przez funkcję w C?

Odpowiedź: Nie.

Ale można szukać alternatyw.

To nie działa, bo tablica jest zmienną lokalną - kończy życie po zakończeniu funkcji.

```c
#include <stdio.h>

int *create_array() {
    int arr[5] = {1, 2, 3, 4, 5};
    return arr;
}

int main() {
    int *array = create_array();
    for (int i = 0; i < 5; i++) {
        printf("%d ", array[i]);
    }
    return 0;
}
```

Rozwiązanie nie jest stabilne - “brak pewności rezerwacyjnej”.

```c
#include <stdio.h>

int *create_array() {
    static int arr[5] = {1, 2, 3, 4, 5};
    return arr;
}

int main() {
    int *array = create_array();
    for (int i = 0; i < 5; i++) {
        printf("%d ", array[i]);
    }
    return 0;
}
```

Jak to zatem zrobić?

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

int *create_array(int size) {
    int *arr = (int *)malloc(size * sizeof(int));
    if (arr == NULL) {
        return NULL;
    }
    for (int i = 0; i < size; i++) {
        arr[i] = i + 1;
    }
    return arr;
}

int main() {
    int size = 5;
    int *array = create_array(size);
    if (array == NULL) {
        printf("Memory allocation failed.\n");
```

Alternatywa - zrobić procedurę, a argument funkcji jest tablicą (wskaźnikiem).

```c
#include <stdio.h>

void create_array(int *arr, int size) {
    for (int i = 0; i < size; i++) {
        arr[i] = i + 1;
    }
}

int main() {
    int array[5];
    create_array(array, 5);
    for (int i = 0; i < 5; i++) {
        printf("%d ", array[i]);
    }
    return 0;
}
```

Napisy (łańcuchy znaków)

Napisy

Napis - ciąg składający się z co najmniej jednego znaku.

Znaki cudzysłowu nie są częścią łańcucha.

Język C nie posiada typu `string`/łańcuchowego. Wszystkie napisy traktowane są jako tablice typu `char`. Ostatnim znakiem w tablicy jest znak `\0`.

Znak a napis

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

int main()
{
    char a = 'q';
    char b[] = "q";
    return 0;
}
```

srtlen a sizeof

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

int main()
{
    char nap1[] = "Hello World";
    char nap2[50] = "Hello World";
    printf("%Iu\n",sizeof nap1);
    printf("%Iu\n",strlen(nap1));
    printf("%Iu\n",sizeof nap2);
    printf("%Iu\n",strlen(nap2));
    return 0;
}
```

Tablica a wskaźnik

```c
#include <stdio.h>
#define NAPIS "jakiś tekst"

int main()
{
    char tab[] = NAPIS;
    const char *wsk = NAPIS;
    printf("adres napisu %p\n", "jakiś tekst");
    printf("adres tab: %p\n", tab);
    printf("adres wsk: %p\n", wsk);
    printf("adres NAPIS-u: %p\n", NAPIS);
    return 0;
}
```

```c
#include <stdio.h>

int main()
{
    char nap1[] = "absddfvjskjf";
    char *nap2 = "oijefj";
    nap1[4] = 'M';
    *(nap1 + 7) = 'M';
    nap2[2]='3'; // czy to zawsze możliwe?
    return 0;
}
```

Kopiowanie napisu

```c
#include <stdio.h>

int main()
{
    char * napis = "ab6sWR";
    char * kopia;
    kopia=napis;
    printf("%s\n",napis);
    printf("%p\n",napis);
    printf("%p\n",&napis);
    printf("%s\n",kopia);
    printf("%p\n",kopia);
    printf("%p\n",&kopia);
    return 0;
}
```

czy można to zrobić notacją tablicową?

Wczytywanie napisów

- `scanf`

https://pl.wikibooks.org/wiki/C/scanf

https://en.cppreference.com/w/c/io/fscanf

- `gets`

https://pl.wikibooks.org/wiki/C/gets

https://en.cppreference.com/w/c/io/gets

- `fgets`

https://pl.wikibooks.org/wiki/C/fgets

https://en.cppreference.com/w/c/io/fgets

Wskaźnik czy tablica?

```c
#include <stdio.h>

int main()
{
    char * slowo;
    scanf("%s",slowo);
    printf("%s\n",slowo);
    return 0;
}
```

```c
#include <stdio.h>

int main()
{
    char slowo[20];
    scanf("%s",slowo);
    printf("%s\n",slowo);
    return 0;
}
```

```c
#include <stdio.h>

int main()
{
    char slowo[5];
    gets(slowo);
    printf("%s\n",slowo);
    puts(slowo);
    return 0;
}
```

- wpisz różnej długości napisy (niektóre ze spacjami)

```c
#include <stdio.h>

int main()
{
    char slowo[5];
    gets_s(slowo, 4*sizeof(char));
    printf("%s\n", slowo);
    puts(slowo);
    return 0;
}
```

- nie działa w każdej konfiguracji

```c
#include <stdio.h>

int main()
{
    char slowo[5];
    fgets(slowo,5,stdin);
    printf("%s\n",slowo);
    puts(slowo);
    fputs(slowo,stdout);
    return 0;
}
```

Różnice?

- `scanf` - do znaku niedrukowanego, reszta do końca linii
- `gets` - mało bezpieczna przy przepełnieniu
- `fgets` - dodaje koniec linii na końcu napisu

Wyświetlanie napisów

- `printf`

https://pl.wikibooks.org/wiki/C/printf

https://en.cppreference.com/w/c/io/fprintf

- `puts`

https://pl.wikibooks.org/wiki/C/puts

https://en.cppreference.com/w/c/io/puts

- `fputs`

https://pl.wikibooks.org/wiki/C/fputs

https://en.cppreference.com/w/c/io/fputs

```c
#include <stdio.h>

int main()
{
    char tekst1[]="abc";
    char tekst2[]= {'a','b','c'};
    char tekst3[]="xyz";
    puts(tekst1);
    puts(tekst2);
    puts(tekst3);
    return 0;
}
```

```c
#include <stdio.h>

int main()
{
    char tekst1[]="abc";
    char tekst2[]= {'a','b','c'};
    char tekst3[]="xyz";
    fputs(tekst1,stdout);
    fputs(tekst2,stdout);
    fputs(tekst3,stdout);
    return 0;
}
```

```c
#include <stdio.h>

int main()
{
    char tekst1[]="abc";
    char tekst2[]= {'a','b','c'};
    char tekst3[]="xyz";
    printf("%s",tekst1);
    printf("%s",tekst2);
    printf("%s",tekst3);
    return 0;
}
```

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

int main()
{
    char buffer[20];
    int a=5;
    int b=7;
    sprintf(buffer,"%5d+%5d=%5d",a,b,a+b);
    printf("%s",buffer);
    return 0;
}
```

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

int main()
{
    char buffer[20];
    int a=5;
    int b=7;
    snprintf(buffer,20*sizeof(char),"%5d+%5d=%5d",a,b,a+b);
    printf("%s",buffer);
    return 0;
}
```

Formaty

https://gist.github.com/pjastr/100599c74e34cd814f671daccd0668

https://gist.github.com/pjastr/37e259e243a7b9d26e9c45d528c84c

https://gist.github.com/pjastr/8c059084a44781c9407ee6ab0a3c5e

https://gist.github.com/pjastr/1285dbb398d7cfed9d372b7ed292d5

https://gist.github.com/pjastr/f59613feebb3900634cbe41658ae2be

Typ `wchar_t`

https://en.wikibooks.org/wiki/C_Programming/wchar.h

https://en.cppreference.com/w/c/language/string_literal

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

int main()
{
    wchar_t buffer[20];
    fgetws(buffer,20,stdin);
    fputws(buffer,stdout);
    return 0;
}
```

Polskie znaki?

```c
#include <stdio.h>
#include <stdlib.h>
#include <wchar.h>
#include <locale.h>

int main()
{
    setlocale(LC_ALL,"");
    wchar_t buffer[20];
    fgetws(buffer,20,stdin);
    fputws(buffer,stdout);
    return 0;
}
```

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

int main()
{
    wchar_t buffer[20];
    wscanf(L"%s",buffer);
    wprintf(L"%s",buffer);
    return 0;
}
```

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

int main()
{
    wchar_t buffer[20];
    int a=3;
    int b=4;
    swprintf(buffer,20*sizeof(wchar_t),L"%d+%d=%d",a,b,a+b);
    wprintf(L"%s",buffer); // na linuxie %ls
    return 0;
}
```

Napisy a funkcje

```
1  int dlugosc(char*napis)
2  {
3      int temp=0;
4      while(*(napis++))
5      {
6          temp++;
7      }
8      return temp;
9  }
```

```
1  int dlugosc2(char napis[])
2  {
3      int temp=0;
4      for(int i=0;napis[i]!='\0';i++)
5      {
6          temp++;
7      }
8      return temp;
9  }
```

czy to możliwe?

```
1  void foo(const char*napis)
2  {
3      *napis='a';
4  }
```

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

char* foo()
{
    return "abc";
}

int main()
{
    printf("%s\n",foo());
    return 0;
}
```

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

char* foo()
{
    char * temp=(char*)malloc(sizeof(char)*10);
    temp[0]='w';
    temp[1]='$';
    temp[2]='a';
    temp[3]='\0';
    return temp;
}

int main()
{
    printf("%s\n",foo());
    return 0;
}
```

Tak nie robimy (!)

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

char* foo()
{
    char temp[10];
    temp[0]='w';
    temp[1]='$';
    temp[2]='a';
    temp[3]='\0';
    return temp;
}

int main()
{
    printf("%s\n",foo());
    return 0;
}
```

Podsumowanie

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

int main()
{
    char tekst1[10]="abcde";
    printf("%Iu\n",sizeof(tekst1));
    printf("%p\n",tekst1);
    printf("%p\n",&tekst1);
    //tekst1="eee";
    //tekst1++;
    tekst1[2]='R';
    printf("%s\n",tekst1);
    return 0;
}
```

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

int main()
{
    char *tekst2="abcde";
    printf("%Iu\n",sizeof(tekst2));
    printf("%p\n",tekst2);
    printf("%p\n",&tekst2);
    tekst2="WERT";
    printf("%s\n",tekst2);
    tekst2++;
    //tekst2[2]='R';
    printf("%s\n",tekst2);
    return 0;
}
```

Funckje znakowe i łańuchowe

https://en.cppreference.com/w/c/string/byte

https://en.cppreference.com/w/c/string/wide

Bibliografia

- Richard Reese, Wskaźniki w języku C, Wydawnictwo Helion 2014.