INFORMATYKA

C

WYDZIAŁ ELEKTROTECHNIKI I INFORMATYKI
ELEKTROTECHNIKA
DR INŻ. MICHAŁ ŁANCZONT

VIII

Zakres wykładu

1. *Tablica znakowa - łańcuchy znakowe*
2. *Parametry wejściowe programu*

```
main(char argv, char * argc[])
```

3. *Biblioteka* `<string.h>` *i* `<ctype.h>`

Tablica znakowa

- Tablica typu char w języku C jest stosowania do przechowywania i przetwarzania ciągów znakowych
- Deklarując tablicę znakową należy tak jak dla tablicy liczbowej określić jej rozmiar
- Przy inicjacji można pominąć informację o rozmiarze tablicy
- Dane przy inicjacji można podawać jako tablicę znakową lub ciąg tekstowy
- Należy pamiętać że zadeklarowana tablica będzie zawierała losowe znaki

Tablica znakowa

```
char tabC[20];

char tabC[] =
{'H','e','l','l','o',',',' ',
,'W','o','r','l','d','!'};

char tabC[]="Hello, World!";
```

Tablica znakowa

```
1   #include <stdio.h>
2   #include <stdlib.h>
3
4
5
6
7
8
9
10
11
12
13
14
15
16
```

C:\WINDOWS\system32\cmd.exe

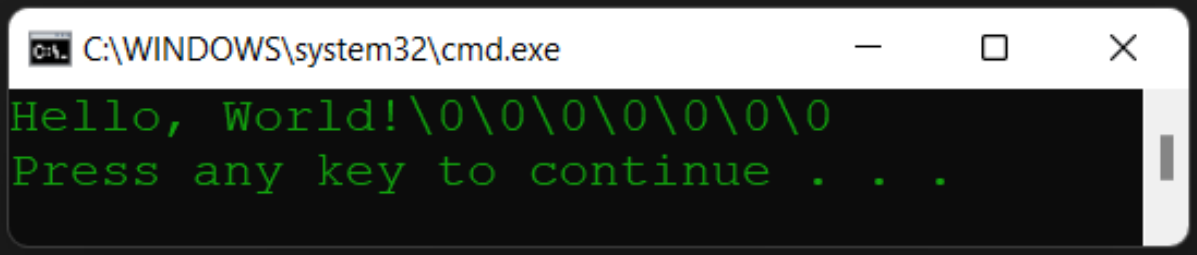
```
Tablica tabC1: Hello, World!
Rozmiar tablicy tabC1: 14
Tablica tabC2: Hello, World!Hello, World!
Rozmiar tablicy tabC2: 13
Tablica tabC3: Hello, World!
Rozmiar tablicy tabC3: 14

Press any key to continue . . .
```

Do prawidłowego przetwarzania danych tablicy znakowej przez funkcje istotne jest aby ostatnim elementem ciągu znakowego był znak „końca ciągu tekstowego” ’\0’

Tablica znakowa

- Za… może
  za… k tablica
  li…
- P… go ”ciąg”
  p… zostaną
  w…

```
1  #include <stdio.h>
2  #include <stdlib.h>
3
4  int main()
5  {
6      char tabC1[20] = "Hello, World!";
7      for(int i=0; i<20;i++){
8          if(tabC1[i]=='\0')
9              printf("\\0");
10         else
11             printf("%c",tabC1[i]);
12     }
13 }
```



Tablica dynamiczna

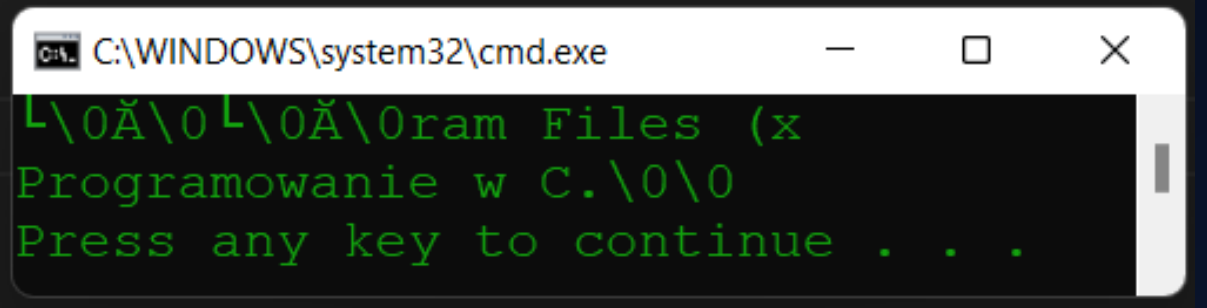
nożna alokować także

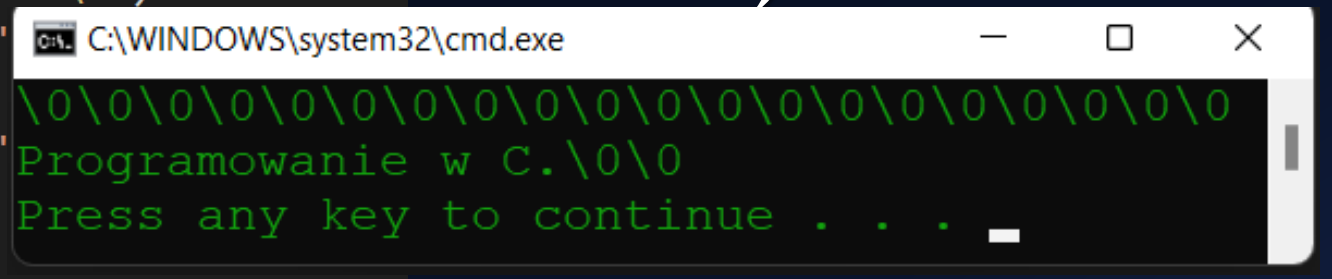
identyczna jak przy
:h

owanej zmiennej

ść każdej komórki na

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

int main()
{
    char *tabC1;
    tabC1 = malloc(20*sizeof(char));
    for(int i=0; i<20;i++){
        if(tabC1[i]=='\0')
            printf("\\0");
        else
            printf("%c",t
    }
    printf("\n");
    tabC1 = "Programowani
    for(int i=0; i<20;i++){
        if(tabC1[i]=='\0')
            printf('
        else
            printf('
    }
    free(tabC1);
}
```





Dane

- Dane do tablicy znakowej można przypisywać:
- Ze standardowego wejścia (klawiatura) cały ciąg tekstowy – `scanf()`, `gets()`
- Przypisując wartości konkretnym komórkom tablicy – `tab[3] = 'c';`
- Czytając znak po znaku ze standardowego wejścia – `scanf()`, `getchar()`
- Jako wynik działania funkcji
- Przy alokacji dynamicznej poprzez przypisanie innej alokowanej tablicy – `tab=tabC;`

Dane

- Funkcja `scanf()` pobiera dane z bufora wejściowego i próbuje je zapisać pod wskazanych adresem do momentu:
  - Niezgodności typów
  - Znaku spacji lub tabulacji
  - Znaku przejścia do nowego wiersza '`\n`'
- Zastosowanie znaku formatującego "`%s`" powoduje że oczekiwany jest ciąg znakowy
- Funkcja `gets()` odczytuje z standardowego wejścia ciąg znakowy do momentu wykrycia znaku przejścia do nowego wiersza '`\n`'

Dane

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

int main()
{
    char *tab;
    tab = (char*) calloc(20,sizeof(char));
    printf("Podaj dane: ");
    gets(tab); //scanf("%s",tab);
    printf("Wprowadzono dane: %s", tab);
    free(tab);
}
```

Dane

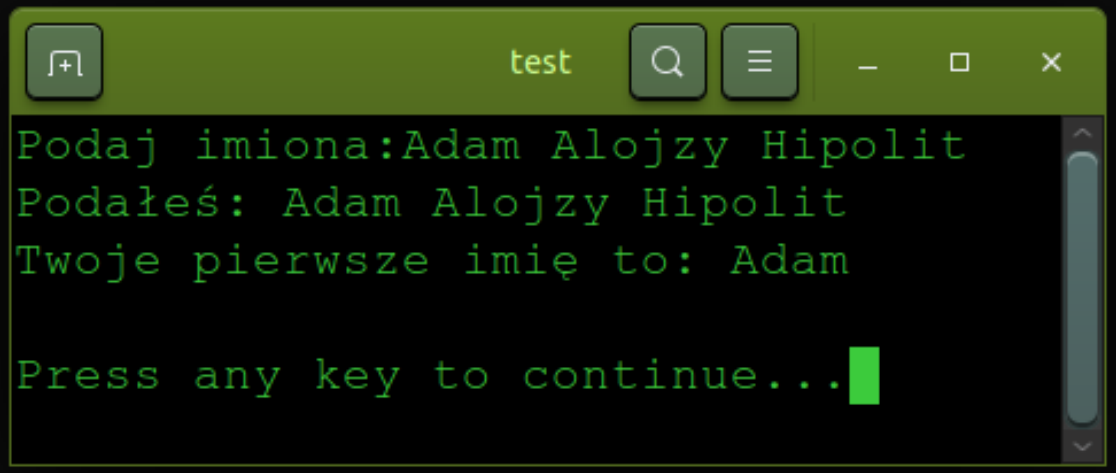
owych z
e funkcji

znaków
wskazanej

ełnieniu”
u na znak

tdin);

```c
#include <stdio.h>

int main()
{
    char buf[1024];
    char imie[30];
    printf("Podaj imiona:");
    fgets(buf, sizeof(buf), stdin);
    printf("Podałeś: %s",buf);
    sscanf(buf,"%s",imie);
    printf("Twoje pierwsze imię to: %s\n",imie);
}
```

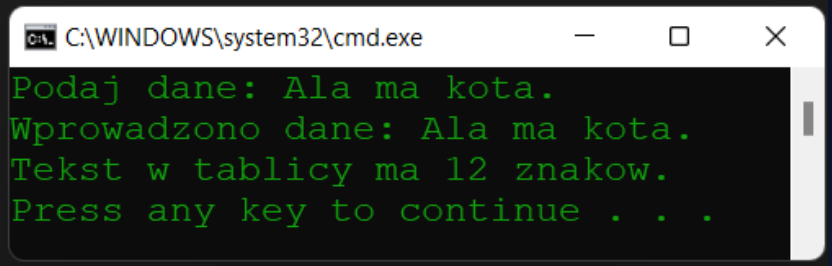


Rozmiar tablicy

- Przy korzystaniu z tablic znakowych w wielu sytuacja istotne jest określenie liczby znaków jakie zostały w tablicy zapisane
- Funkcja `sizeof()` przekaże nam tylko informacje o rozmiarze pamięci zarezerwowanej dla danej tablicy, a nie liczbie zapisanych znaków
- Przy dynamicznej alokacji (`calloc`, `malloc`) funkcja `sizeof()` się nie sprawdzi
- W celu określenia liczby znaków w tablicy (do znaku końca '`\0`') można napisać własną funkcję lub skorzystać z gotowych rozwiązań dostępnych w bibliotece `<string.h>`

Rozmiar tablicy

```c
#include <stdio.h>
#include <stdlib.h>
int sizeoftab(char*);
int main()
{
    char *tab;
    tab = (char*) calloc(20,sizeof(char));
    printf("Podaj dane: ");
    gets(tab); //scanf("%s",tab);
    printf("Wprowadzono dane: %s\n", tab);
    printf("Tekst w tablicy ma %i znakow.",sizeoftab(tab));
    free(tab);
}
int sizeoftab(char*tab)
{
    int i=0;
    do
    {
        i++;
    } while (tab[i]!='\0');
    return i;
}
```



Tablice znakowe 2D

```
1  #include <stdio.h>
2  #include <stdlib.h>
3
4  int main()
5  {
6
7
8
9
10
11
12
13
14
15
16
17
18     printf("Nazywasz sie: %s %s\n", tab2d[0], tab2d[1]);
19     printf("Twoj telefon to: %s", tab2d[2]);
20     for(int i=0;i<3;i++)
21         free(tab2d[i]);
22     free(tab2d);
23 }
```

```
C:\WINDOWS\system32\cmd.exe
Podaj dane osobowe:
Imie: Jan
Nazwisko: Kowalski
Telefon: 600555444
Weryfikacja danych:
Nazywasz sie: Jan Kowalski
Twoj telefon to: 600555444
Press any key to continue . . .
```

leklaruje
tablice

we jest
każdego

jest

każdego

wymaga
ażnika

Parametry wywołania `main()`

z parametrami

*argv[])

**argv)

żone

konsoli wywołując

ą 2D przechowującą

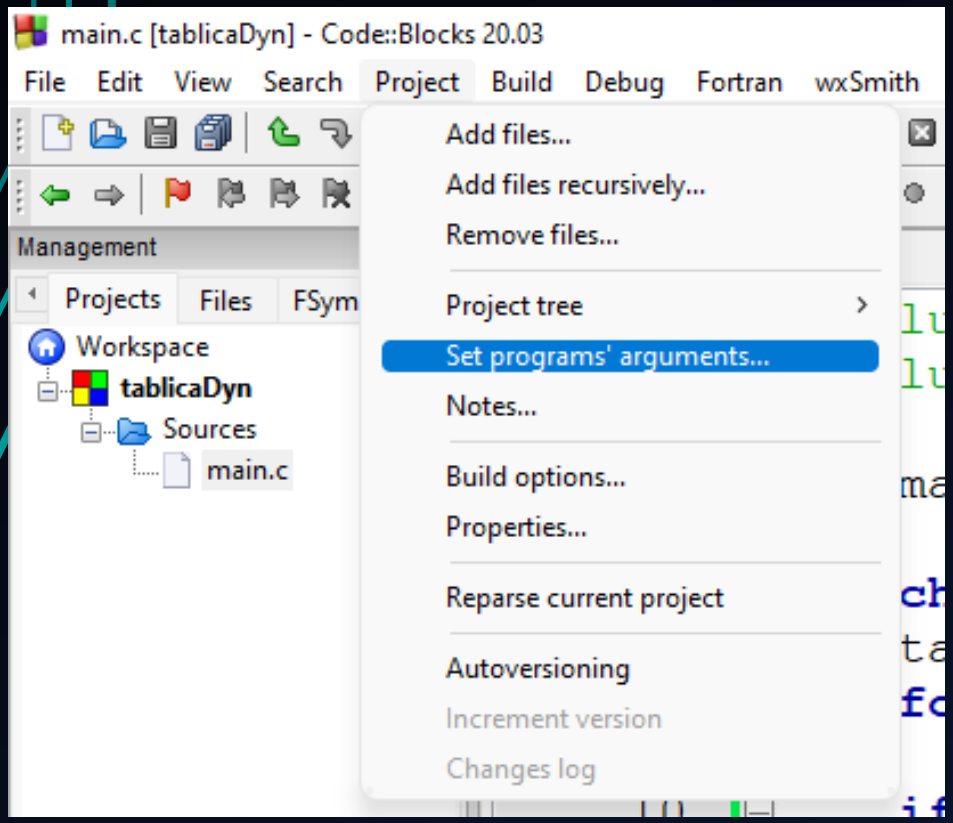
ie o przechowuje

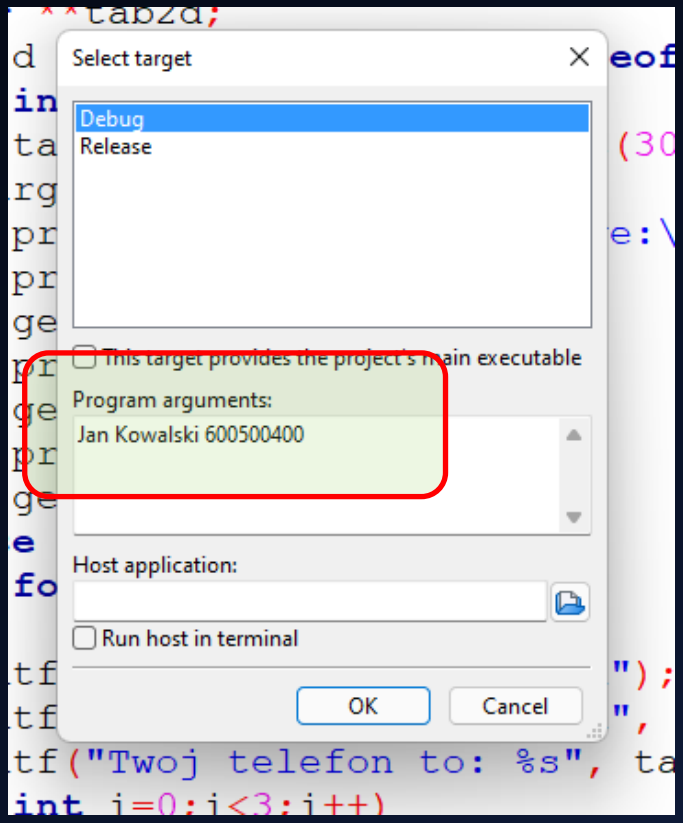
```c
#include <stdio.h>
#include <stdlib.h>

int main(int argc, char **argv)
{
    char **tab2d;
    tab2d = (char**) malloc(3*sizeof(char*));
    for(int i=0; i<3; i++)
        tab2d[i] = (char*) calloc(30,sizeof(char));
    if(argc==1){
        printf("Podaj dane osobowe:\n");
        printf("Imie: ");
        gets(tab2d[0]);
        printf("Nazwisko: ");
        gets(tab2d[1]);
        printf("Telefon: ");
        gets(tab2d[2]);
    }else
        for(int i=0;i<3;i++)
            tab2d[i] = argv[i+1];
    printf("Weryfikacja danych:\n");
    printf("Nazywasz sie: %s %s\n", tab2d[0], tab2d[1]);
    printf("Twoj telefon to: %s", tab2d[2]);
    for(int i=0;i<3;i++)
        free(tab2d[i]);
    free(tab2d);
}
```

```
Windows PowerShell
PS D:\> .\tablica2d.exe Jan Kowalski 600555333
Weryfikacja danych:
Nazywasz sie: Jan Kowalski
Twoj telefon to: 600555333
PS D:\>
```

Code::Blocks

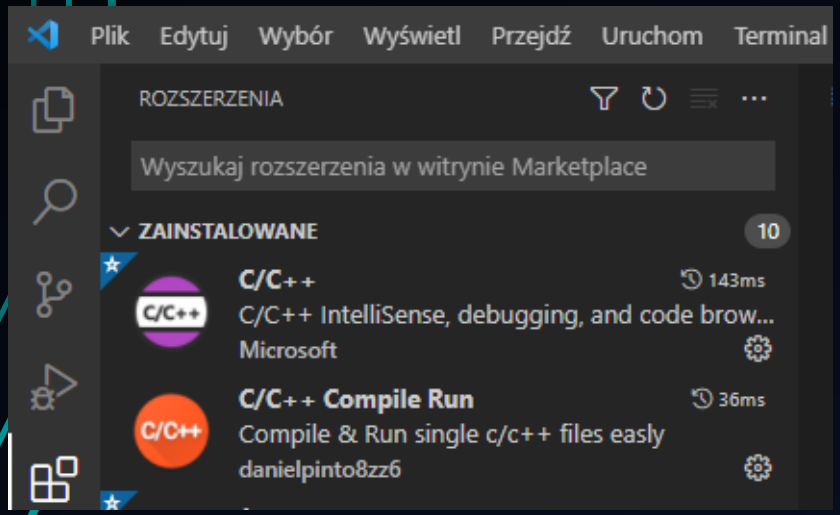
- W środowisku można określić parametry uruchomieniowe kompilowanej aplikacji
- Zakładka ‚Project’
- Pozycja ‚Set programs’ arguments...’

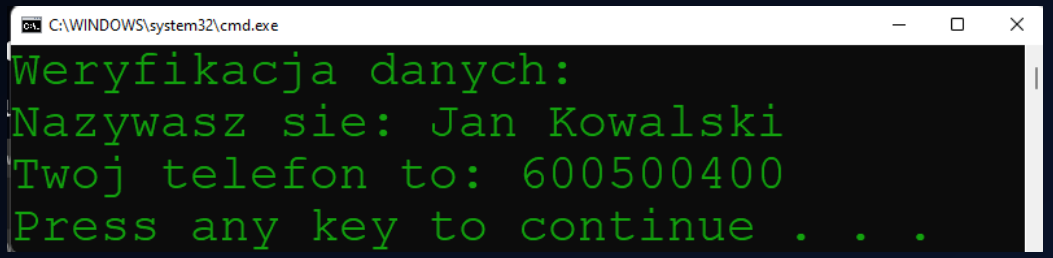


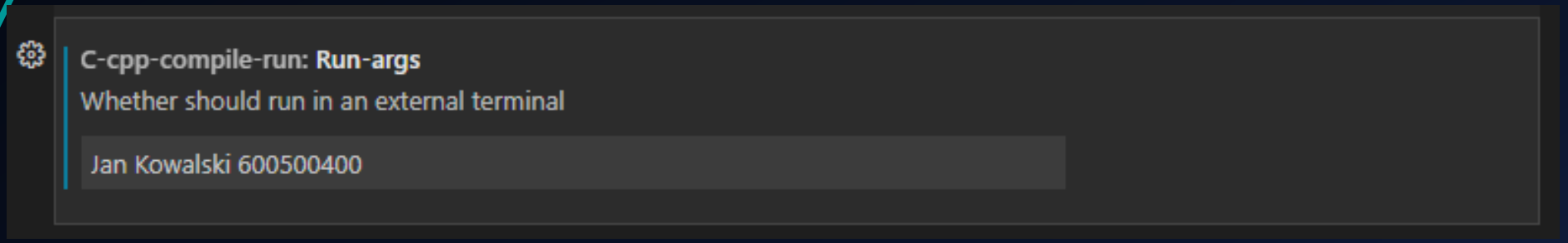


Visual Studio Code

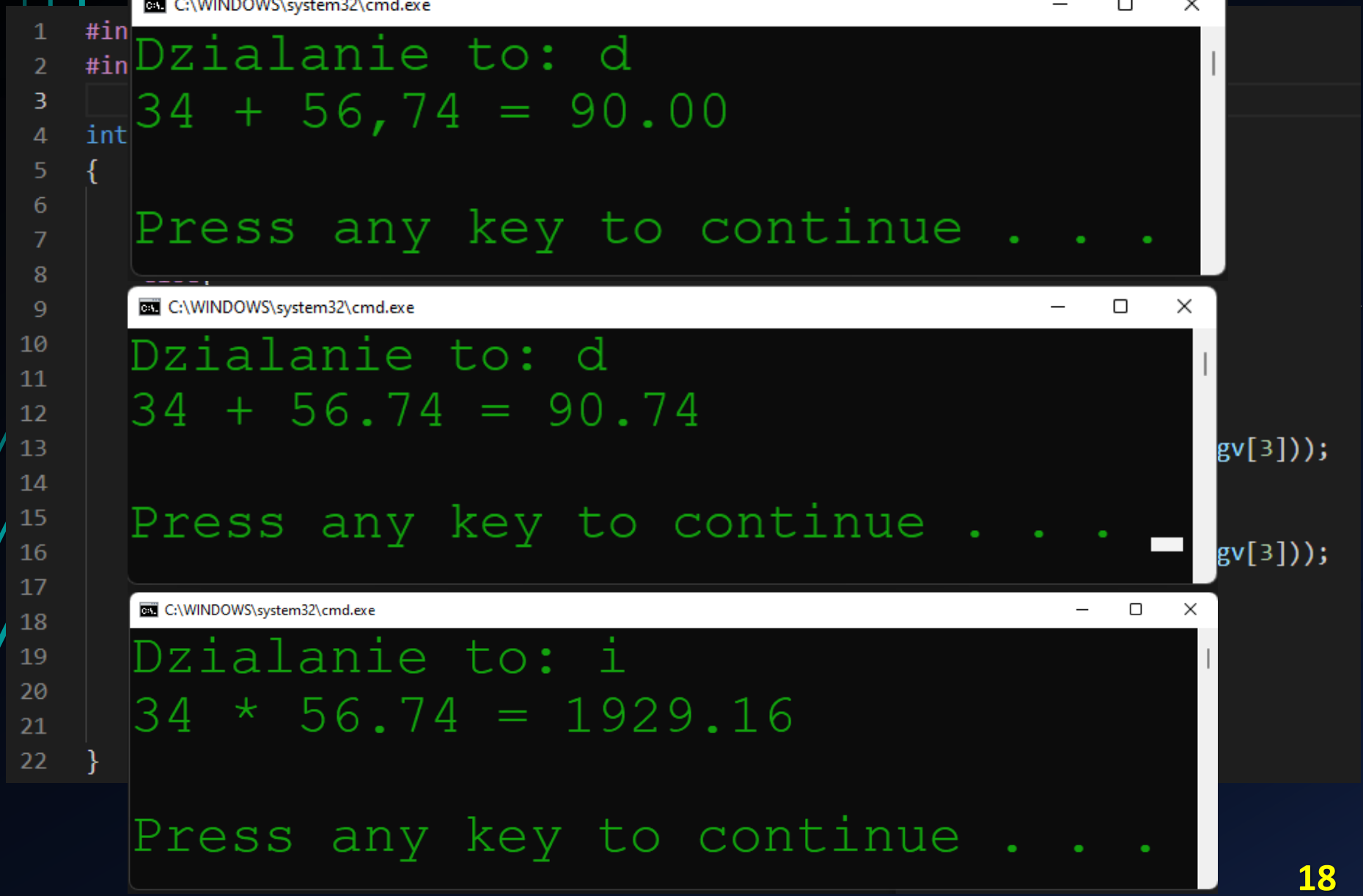
- W ustawieniach wtyczki C/C++ Compiller Run oknie C-cpp-compile-run: Run-args można podać składniki wywołania





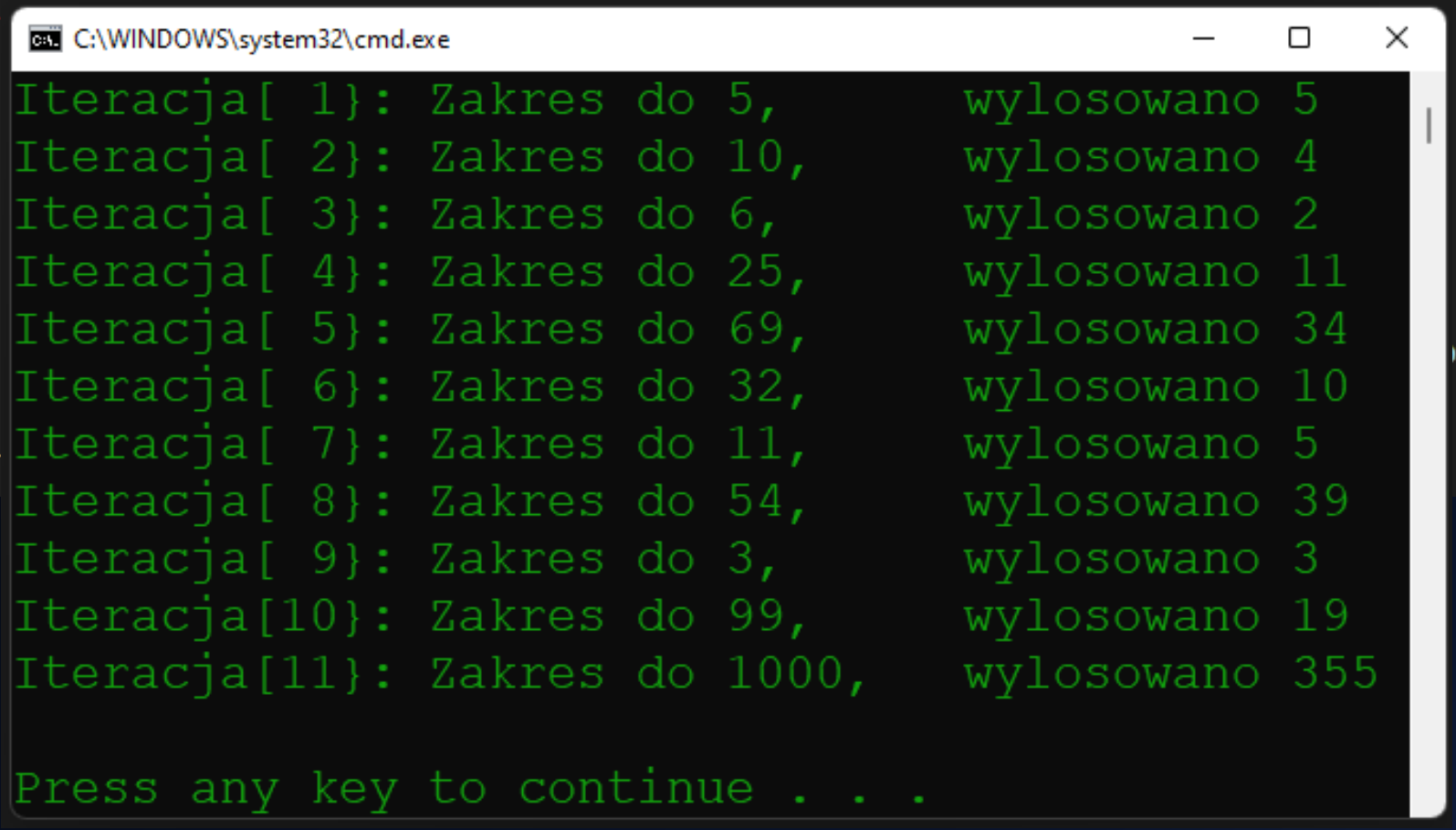


Parametry wywołania `main()`



Strumień tekstowy

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

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

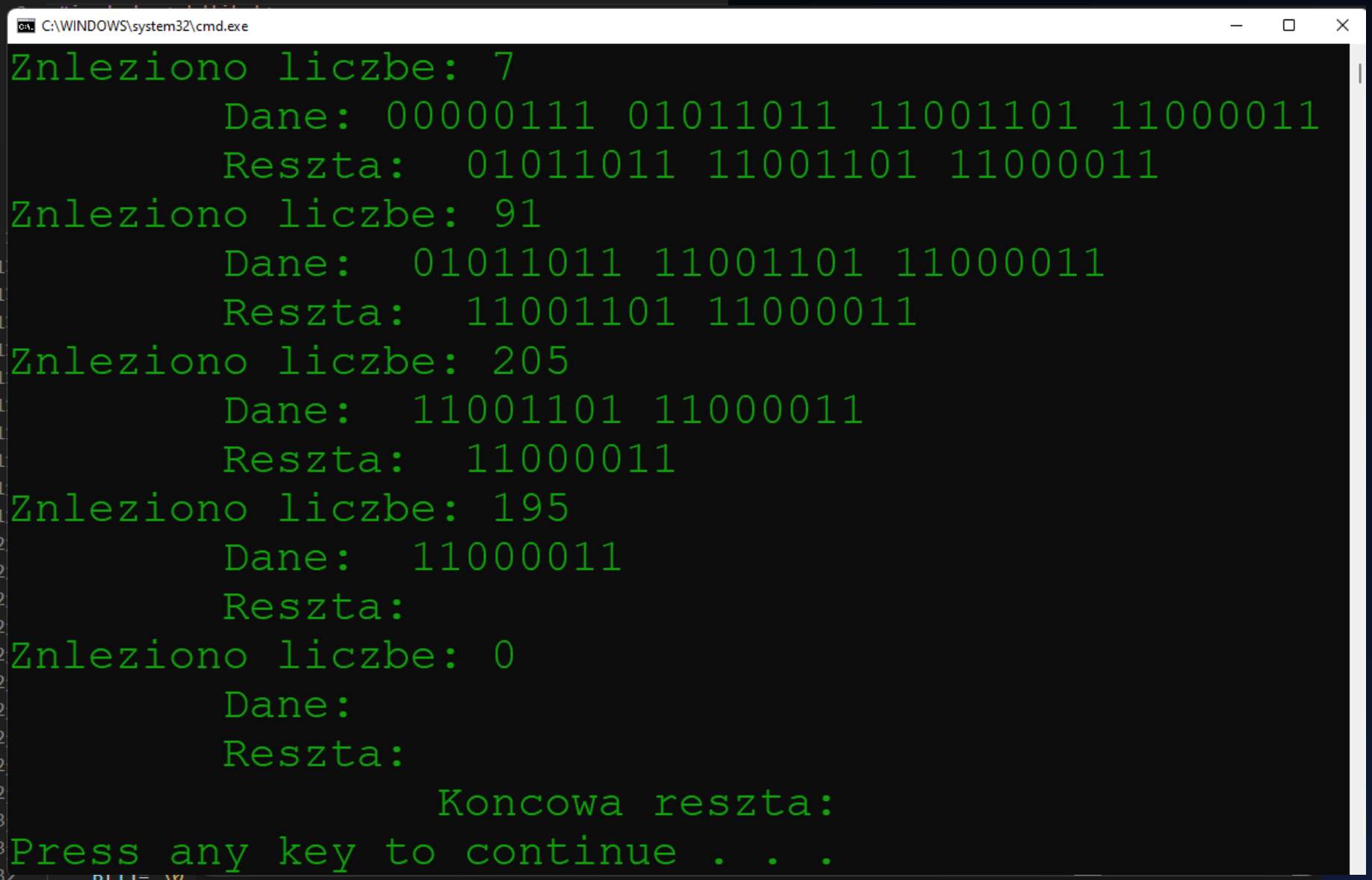


Funkcje konwersji

- Poza funkcjami konwersji bezpośredniej `atoi()`, `atol()` i `atof()` można zastosować funkcję wyszukania i konwersji
- Funkcje `strtol()`, `strtoul()` analizują wejściowy ciąg znakowy i próbuje przekonwertować początkowe znaki na liczbę całkowitą
- Funkcje wykrywają liczbę całkowita zapisaną w notacji określonej współczynnikiem 2-36
- Dane musza zaczynać się od wartości podlegającej konwersji

```
long strtol(char *dane, char **reszta, int baza);
```

Funkcje konwersji



Biblioteka <ctype.h>

- Biblioteka zawiera w większości funkcje testujące podany na wejście znak
  - jest literą alfabetu
  - jest litera dużą lub mała
  - jest cyfrą dziesiętną lub szesnastkową
  - jest znakiem drukowalnym
  - jest spacją
- Funkcje zmiany wielkości znaku (alfabetyczne) z litery małej na dużą i na odwrót
- Funkcje zwracają `0` (`false`) jeżeli warunek testu nie jest spełniony i `1` (`true`) jeżeli jest spełniony

Biblioteka <ctype.h>

`int isalnum(char n)`	czy znak jest cyfrą lub literą alfabetu.
`int isalpha(char n)`	czy znak jest literą alfabetu
`int iscntrl(char n)`	czy znak jest znakiem kontrolnym (od **`0x00`** do **`0x1F`** oraz **`0x7F`**) (*prawy klawisz ALT + kod ASCII znaku z klawiaturze numerycznej*)
`int isdigit(char n)`	czy znak jest cyfrą
`int isgraph(char n)`	czy znak jest znakiem graficznym (np.: ♦)
`int islower(char n)`	czy znak jest małą literą alfabetu
`int isprint(char n)`	czy znak jest znakiem drukowalnym
`int ispunct(char n)`	czy znak jest znakiem drukowalnym ale nie jest znakiem alfanumerycznym ani spacją
`int isspace(char n)`	czy znak jest białym znakiem (spacja, tabulacja)
`int isupper(char n)`	czy znak jest dużą literą alfabetu
`int isxdigit(char n)`	czy znak jest cyfrą szesnastkową
`char tolower(char n)`	konwertuje znak alfabetu z dużej litery na małą
`char toupper(char n)`	konwertuje znak alfabetu z małej litery na dużą

<ctype.h>

```
12      while(dane[n]!='\0'){
13          if(isalnum(dane[n])) test[0]++;
14          if(isalpha(dane[n])) test[1]++;
15          if(iscntrl(dane[n])) test[2]++;
16          if(isdigit(dane[n])) test[3]++;
17          if(isgraph(dane[n])) test[4]++;
18          if(islower(dane[n])) test[5]++;
19          if(isprint(dane[n])) test[6]++;
20          if(ispunct(dane[n])) test[7]++;
21          if(isspace(dane[n])) test[8]++;
22          if(isupper(dane[n])) test[9]++;
23          if(isxdigit(dane[n])) test[10]++;
24          n++;
25      }
```

<ctype.h>

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

int main()
{
```

```
C:\WINDOWS\system32\cmd.exe
Program konertuje znaki na duze litery.
Wprowadz tekst: Testowy tekst do sprawdzenia funkcji konwersji wielk
osci znakow biblioteki <ctype.h>.
Przetworzony tekst zawieral 85 znakow: TESTOWY TEKST DO SPRAWDZENIA
FUNKCJI KONWERSJI WIELKOSCI ZNAKOW BIBLIOTEKI <CTYPE.H>.
Press any key to continue . . .
```

```c
            tekst[i]=toupper(c);
            i++;
            if(i==n){
                n=n*2;
                tekst = (char*)realloc(tekst,n*sizeof(char));
            }
        }
        tekst[i]='\0';
        printf("Przetworzony tekst zawieral %i znakow: %s",i,tekst);
        free(tekst);
}
```

Biblioteka <string.h>

- Biblioteka zawiera szereg funkcji ułatwiających operacje na ciągach znakowych
- Dostępne funkcje:
  - kopiowania
  - łączenia
  - porównania
  - wyszukiwania
  - inne

Inne funkcje

- Określanie liczby znaków w tablicy znakowei `strlen()`

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

int main()
{
    char dane[100] = "Podaj imie i nazwisko: Jan Kowalski.";
    int w = sizeof(dane);
    int r = sizeof(dane)/sizeof(char);
    int lz = strlen(dane);
    printf("Tablica dane ma rozmiar %i bytes, moze przechowywac %i znakow.\n", w,r);
    printf("Do tablicy dane wpisano %i znakow, plus jedne na znak konca ciagu '\\0'.\n",lz);
    printf("Test funkcji memset().\n");
    printf("%s\n",dane);
```

```
C:\WINDOWS\system32\cmd.exe
Tablica dane ma rozmiar 100 bytes, moze przechowywac 100 znakow.
Do tablicy dane wpisano 36 znakow, plus jedne na znak konca ciagu '\0'.
Test funkcji memset().
Podaj imie i nazwisko: Jan Kowalski.
Podaj imie i nazwisko:_____.

Press any key to continue . . .
```

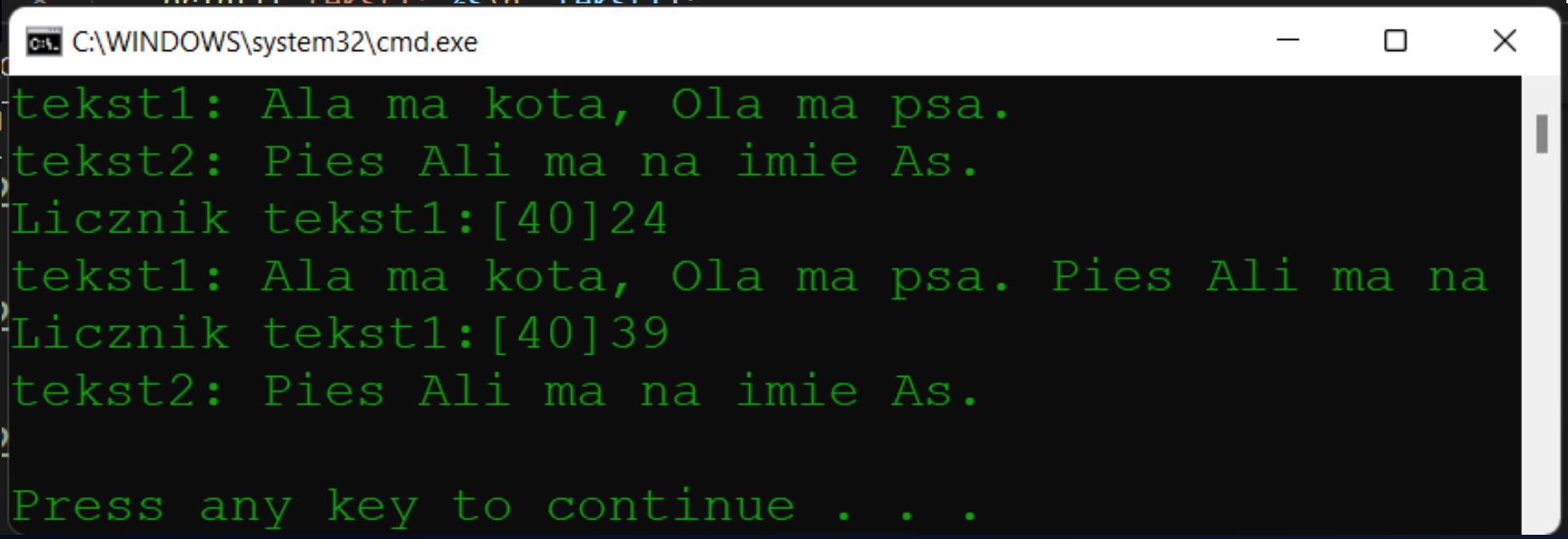
Kopiowanie

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

int main ()
{
  char str[] = "1234567890";
  char str2[25];
  char str3[25];
  puts (str);
  memcpy (str,str+2,8*sizeof(char));
  puts (str);
  puts (str2);
  memmove(str2,str+2,8*sizeof(char));
  puts (str2);
  strcpy(str2,str);
  puts (str2);
  strncpy(str3,str,6);
  puts (str3);
  return 0;
}
```

```
C:\WINDOWS\system32\cmd.exe
1234567890
3456789090
☺
56789090╝§y
3456789090
345678 a

Press any key to continue . . .
```

Łączenie

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

int main()
{
    char tekst2[] = "Pies Ali ma na imie As.";
    char tekst1[40] = "Ala ma kota, Ola ma psa.";
    printf("tekst1: %s\n" tekst1);
```



Porównanie

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

int main()
{
    char text_1[] = "first";
    char text_2[] = "fire ";
```

```
C:\WINDOWS\system32\cmd.exe
Roznica pomiedzy tablicami wynosi 1.
text_1[0]=f : 102        text_2[0]=f : 102
text_1[1]=i : 105        text_2[1]=i : 105
text_1[2]=r : 114        text_2[2]=r : 114
text_1[3]=s : 115        text_2[3]=e : 101
text_1[4]=t : 116        text_2[4]=  : 32
text_1[5]=  : 0          text_2[5]=  : 0

Press any key to continue . . .
```

Wyszukiwanie

- Funkcje wyszukiwania umożliwiają wykonanie szeregu operacji na ciągach tekstowych dzięki identyfikacji pozycji danego znaku lub tablicy w analizowanym ciągu.
  - Wyszukiwanie znaku
  - Wyszukiwanie ciągu znaków
  - Wyszukiwanie znaku z zestawu znaków
- Funkcje zwracają wskaźnik na wyszukany element

Wyszukiwanie znaku

```
char* memchr(char *str, char z, int s);
```

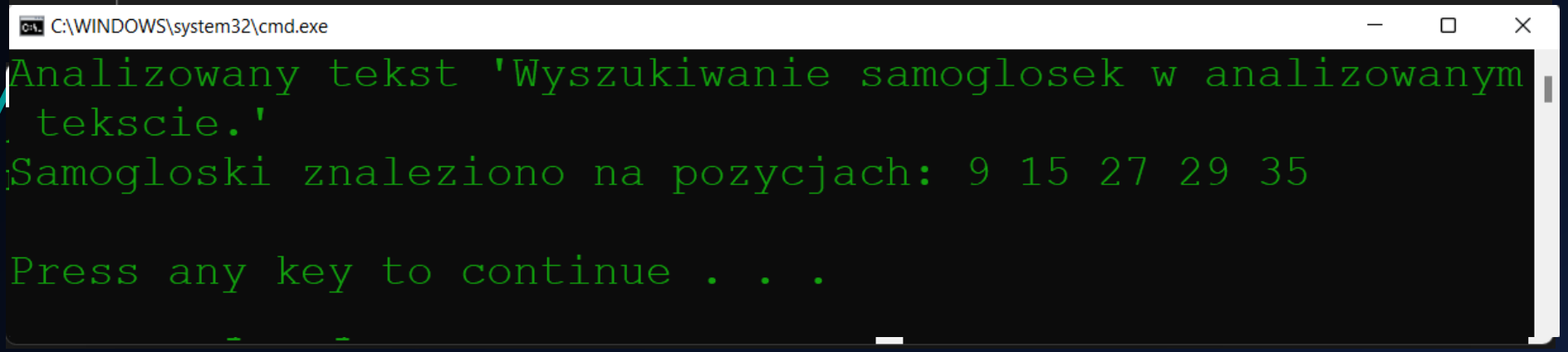
- Funkcja zwraca wskaźnik na znak odnaleziony w obszarze pamięci określonym przez początek tablicy `str` i liczbę bajtów `s`

```
char* strchr(char *str, char z);
```

```
char* strrchr(char *str, char z);
```

- Funkcje przeszukują tablicę `str` w poszukiwaniu znaku `z` od lewego i prawego końca tablicy.
- Funkcje zwracają wskaźnik na odnaleziony znak

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

int main ()
{
  char str[] = "Wyszukiwanie samoglosek w analizowanym tekscie.";
  char * pch;
  printf ("Analizowany tekst '%s'\n",str);
  pch = strchr (str, 'a');
  printf("Samogloski znaleziono na pozycjach: ");
  while (pch != NULL)
  {
    printf ("%i " , pch-str+1);
    pch = strchr(pch+1,'a');
```



Wyszukiwanie znaku z zestawu

Funkcje zwracają informację o odnalezieniu jednego z tablicy podanych znaków w przeszukiwanym ciągu:

- Wskaźnik na odnaleziony znak
- Liczbę znaków jaka jest w ciągu przed odnalezionym znakiem
- Liczbę znaków z zestawu od początku ciągu
- Odnajduje znak z zestawu i zwraca znaki z ciągu występujące po odnalezionym znaku
- Wstawia znak terminalny '`\0`' w miejscu odnalezienia znaku z zestawu

Wyszukiwanie znaku z zestawu

```
char* strpbrk(char *tab, char *zest);
```

- Zwraca wskaźnik do odnalezionego znaku

```
int strcspn(char *tab, char *zest);
```

- Zwraca liczbę znaków od początku ciągu do odnalezionego znaku

```
int strspn(char *tab, char *zest);
```

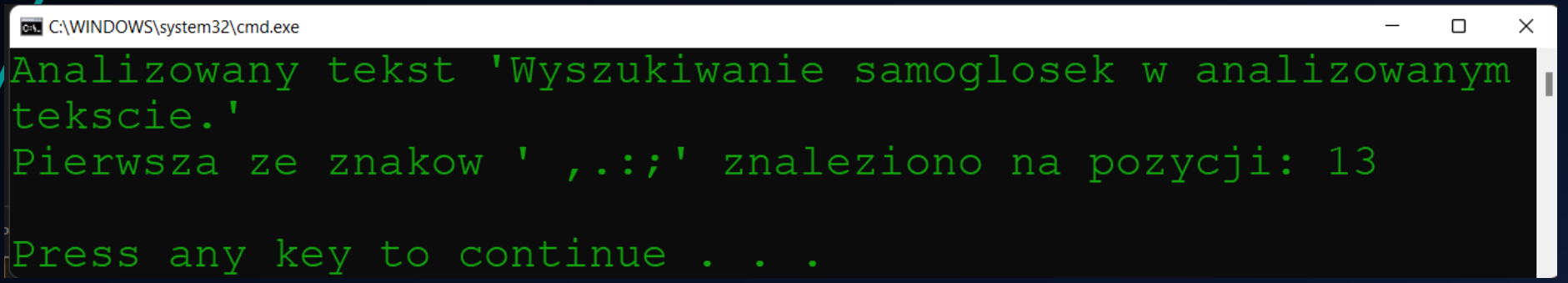
- Zwraca liczbę odnalezionych znaków od początku ciągu do pierwszego znaku który nie występuje w zestawie

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

int main ()
{
  char str[] = "Wyszukiwanie samoglosek w analizowanym tekscie.";
  char key[] = "aeiouy";
  char * pch;
  printf ("Analizowany tekst '%s'\n",str);
  pch = strpbrk (str, key);
  printf("Samogloski znaleziono na pozycjach: ");
  while (pch != NULL)
  {
```

C:\WINDOWS\system32\cmd.exe

```
Analizowany tekst 'Wyszukiwanie samoglosek w analizowanym tekscie.'
Samogloski znaleziono na pozycjach: 2 5 7 9 11 12 15 17 20 22 27 29
31 33 35 37 41 45 46

Press any key to continue . . .
```

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

int main ()
{
  char str[] = "Wyszukiwanie samoglosek w analizowanym tekscie.";
  char key[] = " ,.:;";
  int x;
  printf ("Analizowany tekst '%s'\n",str);
  x = strcspn (str, key);
  printf("Pierwsza ze znakow ' ,.:;' znaleziono na pozycji: %i\n",x+1);
  return 0;
}
```



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

int main ()
{
  char str[] = "123112233442314 56ABCD123456";
  char key[] = "123";
  int x;
  printf ("Analizowany tekst '%s'\n",str);
  x = strspn (str, key);
  printf("Cyfry '123' zajmuja w ciagu: %i miejsc.\n",x);
  return 0;
}
```

```
C:\WINDOWS\system32\cmd.exe
Analizowany tekst '123112233442314 56ABCD123456'
Cyfry '123' zajmuja w ciagu: 9 miejsc.

Press any key to continue . . .
```

Wyszukanie i podmiana

W bibliotece `string.h` dostępna jest funkcja wyszukania i podmiany znaku

Funkcja `strtok()` wyszukuje w wejściowej tablicy znakowej któregokolwiek ze znaków zawartych w tablicy wzorca

W miejscu odnalezionego znaku zostaje wstawiony znak terminalny '`\0`'

Funkcja zwraca wskaźnik na pierwszy znak po wcześniejszym znaku terminalnym, znaku pustym (spacja, tabulacja) lub na początek tablicy

```
char *strtok(char *tab, char *zest);
```

Wyszukanie i podmiana kolejnych znaków w tablicy wymaga modyfikacji polecenia

```
char *strtok(null, char *zest);
```

Wyszukanie i podmiana

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

int main ()
{
  char str[] ="This, a sample string.";
  char * pch;
  printf ("Splitting string \"%s\" into tokens:\n",str);
  int n = strlen(str);
  pch = strtok (str," ");
  while (pch != NULL)
  {
    printf("%i\t",pch-str);
    printf ("%s\n",pch);
```

```
C:\WINDOWS\system32\cmd.exe
Splitting string "This, a sample string." into tokens:
0       This,
6       a
8       sample
15      string.
This,\0a\0sample\0string.

Press any key to continue . . .
```

```c
  }
}
```

Wyszukiwanie ciągu znaków

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

int main()
{
    char dane[] = "Przykladowy ciag z danymi do testow.";
    char zestaw[] = "ciag";
    char *p = strstr(dane,zestaw);
    printf("Szukany ciag znakow: '%s'\n", zestaw);
    printf("Znaleziono w ciagu: '%s'\n",dane);
    printf("Na pozycji: %i\n",p-dane+1);
}
```