```
    sprintf(formal, "%s, %-19s: %6.2f zl\n", nazwisko, imie, wygrana);
    puts(formal);

    return 0;
}
char *wczytaj(char *z, int ile)
{
    char * wynik;
    int i = 0;

    wynik = fgets(z, ile, stdin);
    if (wynik) // wynik rozny od NULL
    {
        while (z[i] != '\n' && z[i] != '\0')
            i++;
        if (z[i] == '\n')
            z[i] = '\0'; // znak nowego wiersza -> znak pusty
        else             // z[i] == '\0'
            while (getchar() != '\n')
                continue;
    }
    return wynik;
}
```

Oto przykładowy wynik działania programu:

```
Podaj swoje imie:
Kmicic
Podaj swoje nazwisko:
Andrzej
Podaj wygrana sume pieniedzy:
450
Kmicic, Andrzej           : 450.00 zl
```

Instrukcja `sprintf()` pobrała dane wejściowe, sformatowała je i zapisała w łańcuchu `formal`.

## Inne funkcje łańcuchowe

Biblioteka ANSI C zawiera ponad 20 funkcji obsługujących łańcuchy. Poniższa lista podsumowuje niektóre z najczęściej używanych:

- `char *strcpy(char * restrict s1, const char * restrict s2);`

  Ta funkcja kopiuje łańcuch (wraz ze znakiem zerowym) wskazywany przez `s2` w miejsce wskazywane przez `s1`. Wartością zwracaną jest `s1`.

- `char *strncpy(char * restrict s1, const char * restrict s2, size_t n);`

  Ta funkcja kopiuje nie więcej niż `n` znaków z łańcucha wskazywanego przez `s2` w miejsce wskazywane przez `s1`. Wartością zwracaną przez `s1`. Nie są kopiowane znaki następujące po znaku zerowym, a jeśli łańcuch źródłowy jest krótszy niż `n` znaków, końcówka łańcucha docelowego jest wypełniana znakami zerowymi. Jeśli łańcuch źródłowy zawiera `n` lub więcej znaków, znak zerowy nie jest kopiowany.

- `char *strcat(char * restrict s1, const char * restrict s2);`

  Łańcuch wskazywany przez `s2` jest dołączany (wraz ze znakiem zerowym) na końcu łańcucha wskazywanego przez `s1`. Pierwszy znak łańcucha `s2` jest umieszczany w miejscu znaku zerowego łańcucha `s1`. Wartością zwracaną jest `s1`.

- `char *strncat(char * restrict s1, const char * restrict s2, size_t n);`

  Nie więcej niż pierwsze `n` znaków łańcucha `s2` jest dopisywane do łańcucha `s1`. Pierwszy znak łańcucha `s2` jest umieszczany w miejscu znaku zerowego łańcucha `s1`. Znak zerowy oraz wszelkie następujące po nim znaki nie są kopiowane z łańcucha `s2`. Do łańcucha wynikowego dodawany jest znak zerowy. Wartością zwracaną jest `s1`.

- `int strcmp(const char * s1, const char * s2);`

  Funkcja zwraca wartość dodatnią, jeśli łańcuch `s1` znajduje się po łańcuchu `s2` według porządku sortowania komputera, wartość `0`, jeśli łańcuchy są identyczne, oraz wartość ujemną, jeśli pierwszy łańcuch znajduje się przed łańcuchem drugim.

- `int strncmp(const char * s1, const char * s2, size_t n);`

  Ta funkcja działa tak samo, jak `strcmp()`, z tym że porównywanie ulega zatrzymaniu po sprawdzeniu `n` znaków (lub w momencie napotkania znaku zerowego).

- `char *strchr(const char * s, int c);`

  Ta funkcja zwraca wskaźnik do pierwszego miejsca w łańcuchu `s` przechowującego znak `c`. (Końcowy znak zerowy jest częścią łańcucha, a więc również on może być obiektem poszukiwań). W przypadku braku znaku funkcja zwraca wskaźnik zerowy.

- `char *strpbrk(const char * s1, const char * s2);`

  Ta funkcja zwraca wskaźnik do pierwszego miejsca w łańcuchu `s1` przechowującego jakikolwiek znak znajdujący się w łańcuchu `s2`. W przypadku braku takiego znaku funkcja zwraca wskaźnik zerowy.

- `char *strrchr(const char * s, int c);`

  Ta funkcja zwraca wskaźnik do ostatniego miejsca w łańcuchu `s` przechowującego znak `c`. (Końcowy znak zerowy jest częścią łańcucha, a więc również on może być obiektem poszukiwań). W przypadku braku znaku funkcja zwraca wskaźnik zerowy.

- `char *strstr(const char * s1, const char * s2);`

  Ta funkcja zwraca wskaźnik do pierwszego wystąpienia łańcucha `s2` w łańcuchu `s1`. W przypadku braku łańcucha funkcja zwraca wskaźnik zerowy.

- `size_t strlen(const char * s);`

  Ta funkcja zwraca liczbę znaków, z wyłączeniem znaku zerowego, z jakiej składa się łańcuch `s`.