

ProgramowanieProceduralne

[Strona główna](#) / [Moje kursy](#) / [PP](#) / [LAB_8](#) / [IS_L7](#)

IS_L7

`int sprintf(char *str, const char *format, ...);` formatowany zapis do stringu

funkcja wysyła sformatowany, według formatu ***format**, wynik do tablicy(stringu) trzymanej przez wskaźnik ***str**

Przykład wykorzystania

```
#include <stdio.h>
#include <string.h>
#define MAX 20
int main(void){
    char imie[MAX];
    char nazwisko[MAX];
    char format[ MAX];
    char napis[2 * MAX + 10];
    double wygrana;

    puts("Podaj swoje imie:");
    gets(imie);
    puts("Podaj swoje nazwisko:");
    gets(nazwisko);
    puts("Podaj wygrana sume pieniedzy:");
    scanf("%lf", &wygrana);

    sprintf(format,"%s, %%ds: %%6.2f z1\n",-19);    //budowanie formatu
    printf ("forma wypisywania %s", format);
    sprintf(napis, format, nazwisko, imie, wygrana);
    puts(napis);
    return 0;}
```

```
void qsort(void * base,size_t num, size_t width,int( * compare )( const void *, const void * ));
```

- `const void *base` - wskaźnik na tablicę, która ma zostać posortowana.
- `size_t num` - liczba elementów w tablicy.
- `size_t width` - liczba bajtów zajmowanych przez jeden element tablicy.
- `int (*compare) (const void *, const void *)` - funkcja porównująca elementy tablicy.

Do argumentów przedmiotowej funkcji trafiają wskaźniki na elementy obecnie porównywane.
Funkcja przekazana jako argument **compare** powinna zwracać następujące wartości:

- < 0** - gdy wartość argumentu pierwszego jest mniejsza od argumentu drugiego;
- = 0** - gdy wartość argumentu pierwszego jest równa wartości argumentu drugiego;
- > 0** - gdy wartość argumentu pierwszego jest większa od argumentu drugiego.

1.
(2) Proszę napisać program, który wypisze:

```
FORMAT %.0lf ->3
FORMAT %.2lf ->3.14
FORMAT %.4lf ->3.1416
FORMAT %.6lf ->3.141593
```

```
#include .....
#include .....
#include .....
int main(){
char pi_form[7]=.....;
int i;
for (i=0; i<7; i+=2){
    .....(pi_form, ".....",i); //zbudowanie łańcucha formatującego
    printf("FORMAT ..... ->",pi_form); //wypisanie łańcucha formatującego
    .....(....., "\n");//doklejenie znaku końca linii
    printf(.....,M_PI); //M_PI stała z biblioteki math.h
}
}
```

2.

(2) Proszę napisać funkcję **zero** znajdującą miejsce zerowe **m_z** funkcji **f** metodą bisekcji z dokładnością **eps**

```
int zero(double (*f)(double), double a, double b, double* m_z, double eps);
```

- ***f** wskaźnik do funkcji
- **a, b** - początek i koniec przedziału
- **m_z** miejsce zerowe
- **eps** - dokładność
- funkcja zwraca 0 gdy nie znajdzie miejsca zerowego, 1 gdy znajdzie

Proszę przetestować dla funkcji

- **sin(x)** w przedziale <-1, 1>
- **-x²+3*x+11** w przedziale <0, 8 > (5.14)
- **-x²+3*x+11** w przedziale <-1, 1 > (nie ma)

3.

(5) Proszę uzupełnić program sortującą tablicę liczb rzeczywistych, jednowymiarową tablicę stringów oraz dwuwymiarową tablicę napisów z wykorzystaniem funkcji **qsort**

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

..... double_cmp(....., .....); //komparator dla double
..... cstring_cmp(....., .....); //komparator dla stringow
..... tabchars_cmp(....., .....); //komparator dla tablicy napisow

void print_double_array(double*,int); //wypisywanie tablicy double
void print_cstring_array(char**,int); //wypisywanie tablicy string'ow
void print_tabchars_array(char(*)[10],int); //wypisywanie tablicy napisow

void sort_double_example() //sortowanie tablicy double
{ double numbers[] = { 7.4, 1.3, 14.5, 0.1, -1.0, 2.3, 1,2, 43.0, 2.0, -4.7, 5.8 };
  size_t numbers_len = _____; //okreslenie dlugosci tablicy
  puts("*** Double sorting...");
  print_double_array(_____, _____); //wypisanie tablicy double przed sortowaniem
  qsort(_____, _____, _____, _____); //sortowanie tablicy double funkcja qsort
  print_double_array(_____, _____); //wypisanie tablicy double po sortowaniu
}

void sort_cstrings_example() //sortowanie tablicy stringow
{ char *strings[] = { "Zorro", "Alex", "Celine", "Bill", "Forest", "Dexter"};
  size_t strings_len = _____; //okreslenie dlugosci tablicy
  puts("*** String sorting...");
  print_cstring_array(_____, _____); //wypisanie tablicy stringow przed sortowaniem
  qsort(_____, _____, _____, _____); //sortowanie tablicy stringow funkcja qsort
  print_cstring_array(_____, _____); //wypisanie tablicy stringow sortowaniu
}

void sort_tabchars_example() //sortowanie tablicy stringow
{ char tab_char[][10] = { "Zorro", "Alex", "Celine", "Bill", "Forest", "Dexter"};
  size_t tab_char_N = _____; //okreslenie ilosci napisow w tablicy
  puts("*** table sorting...");
  print_tabchars_array(_____, _____); //wypisanie tablicy napisow przed sortowaniem
  qsort(_____, _____, _____, _____); //sortowanie tablicy stringow funkcja qsort
  print_tabchars_array(_____, _____); //wypisanie tablicy napisow sortowaniu
}

// MAIN program (wywołanie funkcji sortujacych)
int main() {
  sort_double_example();
  sort_cstrings_example();
  sort_tabchars_example();
return 0; }
```

4.
(4) W programie mamy zdefiniowane następujące funkcje :

```
double fun0(double x) { return log(x); }
double fun1(double x) { return x*x; }
double fun2(double x) { return sin(x); }
double fun3(double x) { return cos(x); }
```

oraz tablicę stringów zawierającą nazwy funkcji - kolejność odpowiada kolejności funkcji

```
..... nazwy[]{"log", "pow","sin","cos"};
```

- Proszę stworzyć 5-cio elementową tablicę wskaźników do funkcji **TAB_FUN**, tak aby można było dokonać podstawień - kolejność zgodna z kolejnością nazw w tablicy **nazwy** :

```
TAB_FUN[0] = fun0;
TAB_FUN[1] = fun1;
TAB_FUN[2] = fun2;
TAB_FUN[3] = fun3;
TAB_FUN[4] = NULL;
```

- Proszę stworzyć wskaźnik **wsk_fun** taki, żeby można było poprawnie podstawić:

```
wsk_fun = TAB_FUN;
```

- Przy użyciu wskaźnika **wsk_fun** oraz wskaźnika **str** proszę wypisać wartości funkcji z tablicy **TAB_FUN** dla argumentów z tablicy **data** . Wypisanie ma być zrealizowane według podanego poniżej wzoru - najlepiej bez dodatkowych zmiennych poza **wsk_fun** oraz **str**

```
double data[8] = { 0.5, 1.0, 1.5, 2.0, 2.5, 3.0, 3.5, 4.0 };
```

```
..... str = nazwy;
..... { //pętla po wskaźnikach do funkcji
for (i = 0; i < 8; i++){ //pętla po argumentach
    printf(" ", ....., ....., .....);
printf ("\n");
.....;
.....;
}
```

Wynik wypisania :

log(0.5)= -0.6931 log(1.0)= 0.0000 log(1.5)= 0.4055 log(2.0)= 0.6931 log(2.5)= 0.9163 log(3.0)= 1.0986 log(3.5)= 1.2528 log(4.0)= 1.3863

pow(0.5)= 0.2500 pow(1.0)= 1.0000 pow(1.5)= 2.2500 pow(2.0)= 4.0000 pow(2.5)= 6.2500 pow(3.0)= 9.0000 pow(3.5)= 12.2500 pow(4.0)= 16.0000

sin(0.5)= 0.4794 sin(1.0)= 0.8415 sin(1.5)= 0.9975 sin(2.0)= 0.9093 sin(2.5)= 0.5985 sin(3.0)= 0.1411 sin(3.5)= -0.3508 sin(4.0)= -0.7568

cos(0.5)= 0.8776 cos(1.0)= 0.5403 cos(1.5)= 0.0707 cos(2.0)= -0.4161 cos(2.5)= -0.8011 cos(3.0)= -0.9900 cos(3.5)= -0.9365 cos(4.0)= -0.6536





- Następnie proszę napisać funkcję

```
.....find_max(....., .....);
```

zwracając taki **wskaźnik**, który dla wczytanego argumentu typu **double** pozwoli wypisać wartość funkcji (korzystając z tablicy **TAB_FUN**) oraz nazwę funkcji (korzystając z tablicy **nazwy**), która ma największą wartość - proszę pamiętać, że nazwy funkcji są zapisane w tablicy **nazwy** w tej samej kolejności co **wskaźniki** do funkcji (odpowiadającym nazwom) w tablicy **TAB_FUN**.(nazwy[0] -->log TAB_FUN[0]-->log; nazwy[1] -->pow TAB_FUN[1]-->pow; ...)

Wywołanie `wsk_max=find_max (TAB_FUN, 0.05);`
Wypisaniu `printf ("dla x= %f największa wartosc ma %s, ktora wynosi %.3f\n", var, nazwy[//wykorzystac wsk_max.....], (//wykorzystac_wsk_max.....)(var));`
dla x=0.05 największa wartosc ma cos, ktora wynosi 0.998

Status przesłanego zadania

Status przesłanego zadania	Przesłane do oceny
Stan oceniania	Nieocenione
Termin oddania	poniedziałek, 20 kwietnia 2020, 14:25
Pozostały czas	8 min. 27 sek.
Ostatnio modyfikowane	poniedziałek, 20 kwietnia 2020, 14:16
Przesyłane pliki	<div><div><div>-1.c</div><div>-2.c</div><div>-3.c</div><div>-4.c</div></div><div><div>20 kwietnia 2020, 14:16</div><div>20 kwietnia 2020, 14:16</div><div>20 kwietnia 2020, 14:16</div><div>20 kwietnia 2020, 14:16</div></div></div>

Komentarz do przesłanego zadania

 [Komentarze \(0\)](#)

Edytuj zadanie

Usuń zadanie

Możesz nadal zmieniać złożone zadanie.



Platforma e-Learningowa obsługiwana jest przez:
Centrum e-Learningu AGH oraz Uczelniane Centrum Informatyki AGH

Podsumowanie zasad przechowywania danych
Pobierz aplikację mobilną