Imię i nazwisko Marek Kubicki	Kierunek Informatyka techniczna ITE	Rok studiów i grupa 1 rok grupa 5
Data zajęć 8.12.2022	Numer i temat sprawozdania Lab 9. – Wskaźniki.	

Cel:

- Opanowanie podstaw wykorzystania wskaźników w C.

Przebieg zajęć:

Utworzyłem katalog roboczy lab_9 i skopiowałem do niego pliki wskaźniki_simple.c
Następnie zacząłem wykonywać polecenia zawarte w pliku PP_L09_Wskazniki.pdf

Należy unikać arytmetyki wskaźników ponieważ można może to zaowocować wskaźnikiem wskazującym w niewłaściwe miejsce w pamięci co często powoduje błędne działanie programu, lub nawet przedwczesne przestanie działania.

Na początku zadeklarowałem i przypisałem wartości do zmiennych i wskaźników.

```
skr1.asc  wskazniki_simple.c
m@m-VirtualBox:~/Lab_9$ pwd
/home/m/Lab_9
```

```
  int i = 1;
  int * wskaznik_do_int = &i; /

  printf("\nprzed wywołaniem prostej funkcji\n");
  printf("\nwartość zmiennej typu int i jej adres:    i = %d, &i = %lu \n",
   i, &i);
  printf("to samo co powyżej, za pomocą wskaźników: i = %d, &i = %lu \n",
   *wskaznik_do_int, wskaznik_do_int);

  double pi = 3.14;
  double * double_p = NULL;
  printf("\nwartość zmiennej pi: %lf, adres zmiennej pi: %lu\n", pi, &pi);
  printf("wskaźnik double_p zainicjowany wartością NULL:      %lu\n", do
uble_p);
  double_p = &pi;
  printf("wskaźnik double_p po podstawieniu wartości &pi:     %lu\n", do
uble_p);

  double_p++;
  printf("wskaźnik double_p po wykonaniu operacji double_p++: %lu\n", do
uble_p);
double_p--;
```

```
przed wywołaniem prostej funkcji

wartość zmiennej typu int i jej adres:    i = 1, &i = 140735132172108
to samo co powyżej, za pomocą wskaźników: i = 1, &i = 140735132172108

wartość zmiennej pi: 3.140000, adres zmiennej pi: 140735132172112
wskaźnik double_p zainicjowany wartością NULL:      0
wskaźnik double_p po podstawieniu wartości &pi:     140735132172112
wskaźnik double_p po wykonaniu operacji double_p++: 140735132172120
```

Potem wywołałem funkcję podając jako argumenty wskaźniki i wykonałem podobne operacje jak przed wywołaniem funkcji

```
printf("\n\nwewnątrz prostej funkcji: \n");
  printf("wartość zmiennej typu int i jej adres: i = %d, &i = %lu \n",
     *wskaznik_do_int, wskaznik_do_int);
  printf("\nwartość zmiennej pi: %lf, adres zmiennej pi: %lu\n", *double_p,
double_p);
*wskaznik_do_int++;
*double_p++;
  printf("\nZmienne po operacji ++: \n");
printf("wartość zmiennej typu int i jej adres: i = %d, &i = %lu \n",
     *wskaznik_do_int, wskaznik_do_int);
  printf("\nwartość zmiennej pi: %lf, adres zmiennej pi: %lu\n", *double_p,
double_p);
wewnątrz prostej funkcji:
wartość zmiennej typu int i jej adres: i = 1, &i = 140735132172108

wartość zmiennej pi: 3.140000, adres zmiennej pi: 140735132172112

Zmienne po operacji ++:
wartość zmiennej typu int i jej adres: i = 1374389535, &i = 140735132172112

wartość zmiennej pi: 0.000000, adres zmiennej pi: 140735132172120
```

Po zakończeniu działanie funkcji ponownie wypisałem wartości zmiennych i ich adresy

```
rintf("\n\nwewnątrz prostej funkcji: \n");
  printf("wartość zmiennej typu int i jej adres: i = %d, &i = %lu \n",
     *wskaznik_do_int, wskaznik_do_int);
  printf("\nwartość zmiennej pi: %lf, adres zmiennej pi: %lu\n", *double_p,
double_p);
*wskaznik_do_int++;
*double_p++;
  printf("\nZmienne po operacji ++: \n");
printf("wartość zmiennej typu int i jej adres: i = %d, &i = %lu \n",
     *wskaznik_do_int, wskaznik_do_int);
  printf("\nwartość zmiennej pi: %lf, adres zmiennej pi: %lu\n", *double_p,
double_p);
```

```
po powrocie z prostej funkcji:
wartość zmiennej typu int i jej adres:    i = 1, &i = 140735132172108
to samo co powyżej, za pomocą wskaźników: i = 1, &i = 140735132172108

wartość zmiennej pi i jej adres:           pi = 3.140000, &pi = 140735132172112
to samo co powyżej, za pomocą wskaźników:  pi = 3.140000, &pi = 140735132172112
```

Zmiana adresu na którego wskazywały wskaźniki spowodowało wypisanie wartości znacząco różniących się od tych podanych wcześniej, jednak poza funkcją zmienne i wskaźniki zachowały swoje pierwotne wartości.

Następnie skopiowałem plik równanie_kwadratowe.c i skrypt go uruchamiający z kotalogu z zajęć nr. 5, potem wykonałem dalsze polecenia z wcześniej wspomnianego pliku pdf.

```
sh-4.2$ pwd
/home/METAL/markubi2/lab_9
sh-4.2$ ls
rownanie_kwadratowe.c  skr1  skr2  wskazniki_simple.c
```

Stworzyłem funkcję która zawierała w sobie obliczenia oryginalnie zawarte w main. Wywołałem wszystkie potrzebne mi wartości w funkcji main i stworzyłem do nich wskaźniki

```
  double x1, x2, z1, z2, czy_d;
  double * wsk_x1 = &x1;
  double * wsk_x2 = &x2;
  double * wsk_z1 = &z1;
  double * wsk_z2 = &z2;
  int czy_d=0;
  int * wsk_czy_dok = &czy_dok;
  int nie_pop =0;
  int * wsk_nie_pop = &nie_pop;
```

Następnie wywołałem funkcję wykonującą obliczenia, po wykonaniu obliczeń przekazywała ona odpowiednie wartości do main a w nim wypisywany był wynik.

```
czy_d = funkcja(a, b, c, wsk_x1, wsk_x2, wsk_z1, wsk_z2, wsk_nie_pop,
wsk_czy_dok);
```

Przykładowe zwracanie wartości:

```
*wsk_x1 = (-b)/(2*a);
*wsk_z1 = (temp/(2*a));
*wsk_x2 = (-b)/(2*a);
*wsk_z1 = (temp/(2*a));
czy_d = -1;
```

Wypisywanie wyniku:

```
  if(czy_d == -1)
  {
  printf("Dwa pierwiastki zespolone: x1 = %.12le + i%.12le, x2 = %.12le -
i%.12le \n",x1, z1, x2, z2);
```

```
}
else if(czy_d == 0)
{
printf("Jeden pierwiastek rzeczywisty: x = %.12le\n", x1 );
}
else
{
printf("Dwa pierwiastki rzeczywiste: x1 = %.12le, x2 = %.12le\n",
       x1, x2 );
if(czy_dok==0)
{
printf("Pierwiastki sa dokladne\n");
}
else
{
printf("Pierwiastki nie sa dokladne\n");
}
```

```
Podaj parametr a: 1
Podaj parametr b: 4
Podaj parametr c: 3
Dwa pierwiastki rzeczywiste: x1 = -3.000000000000e+00, x2 = -1.000000000000e+00
Pierwiastki sa dokladne
```

Wnioski

Wskaźniki są użytecznym narzędziem pozwalającym nam na łatwe przekazywanie wielu wartości różnego typu z funkcji do miejsca w którym była wywołana. Dzięki temu można tworzyć bardziej skomplikowane funkcje zwracające wiele różnych wartości.