Laboratorium Informatyki I – LAB 9 - Struktury

1. Wprowadzenie

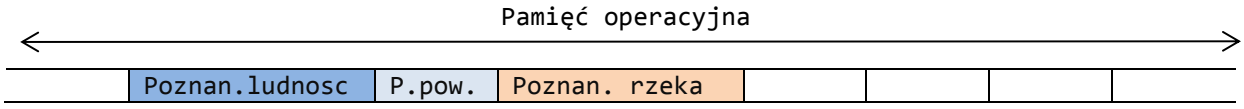
Struktura lub **rekord** - to złożony typ danych występujący w wielu językach programowania, grupujący logicznie powiązane ze sobą dane różnego typu w jednym obszarze pamięci. Składowe struktury - pola - są etykietowane, tj. mają swoje unikatowe nazwy; poprzez podanie nazwy otrzymuje się dostęp do danego pola.

Podobnie jak **tablica,** struktura jest typem danych pozwalającym przechowywać pewne wartości pod jedną nazwą. Pola struktury możemy sami zdefiniować. Przeanalizujmy poniższy przykład:

Struktura w języku C

```
/* deklaracja */
struct miasto
{
    long ludnosc;
    int powierzchnia; //km^2
    char* rzeka;
};

/* definicja */
struct miasto Poznan;
Poznan.ludnosc = 550000;
Poznan.powierzchnia = 262;
Poznan.rzeka = (char*)malloc(sizeof(char)*6);
strncpy(Poznan.rzeka, "Warta\0", 6);
```

Struktura miasto pozwala nam tworzyć nowe rekordy typu miasto (np. Warszawa, Poznan, itd) i w nich zapisać informacje związane z danym miastem (ludność, powierzchnia i rzeka np. Poznan.ludnosc = 550000;) W ten sposób dane są przejrzyście pogrupowane. Podobnie jak w przypadku tablicy, elementy zdefiniowanej struktury są przechowywane w pamięci komputera obok siebie. Schematycznie można to przedstawić tak:



2. Pierwsze kroki

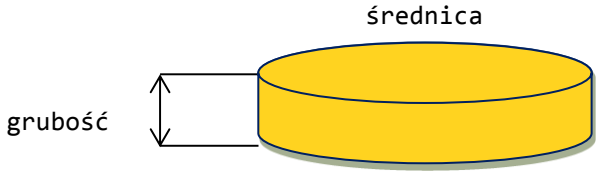
W języku C struktury deklarujemy po dyrektywach preprocesora (include, define). Przykład:

```
#include <stdio.h>
#include <stdlib.h>
#include <math.h>
#define Pi 4*atan(1.)

struct probka {          /*deklaracja*/
   double masa;
   double grubosc;
   double srednica;
};                        /*pamietaj o sredniku tutaj*/

void main(){
  struct probka A;       /*definicja*/
```

```
  printf("Podaj dane probki A: masa[g], grubosc[cm], srednica[cm]\n");
  scanf("%lf%lf%lf",&A.masa, &A.grubosc, &A.srednica);

  /*obliczenie gestosci*/

  double gestosc = Pi*(A.srednica/2.)*(A.srednica/2.)*A.grubosc;
  gestosc = A.masa/gestosc;

   printf("Gestosc probki A wynosi %lf\n ",gestosc);
   system("pause");
   }
```

Ćwiczenie 2.1

Modyfikując powyższy kod, napisz program, który będzie przyjmował dane 3 próbek A, B, C o kształcie walca i wyznaczał gęstość każdej z nich. Dane próbek będą grupowane za pomocą struktury `probka`. Dane próbek B i C powinny być także wprowadzane z klawiatury.



Ćwiczenie 2.2 - Struktura jako argument funkcji

Zmodyfikuj dalej kod nad którym pracujesz. Napisz funkcję double gestosc(struct probka X), która przyjmuje jako argument strukturę z danymi dowolnej próbki i zwraca jej gęstość. Funkcja ta powinna być zadeklarowana poniżej deklaracji struktury. Wykorzystaj napisaną funkcję w funkcji main do liczenia gęstości próbek. Zachowaj gdzieś kod programu, bo będziemy go jeszcze używać.

3. Zwracanie struktury przez funkcję

Przeanalizuj poniższy kod. Funkcja fun oblicza sumę oraz różnicę 2 liczb typu int i wynik zwraca jako strukturę „twoints". Zwróć uwagę na przypisanie: `A = fun(4,3);` które umożliwia zwrócenie danych do funkcji main.

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

struct twoints {
   int i;
   int j;
};

struct twoints fun(int a, int b) { //funkcja  typu struct twoints
struct twoints L;
       L.i=a+b;
       L.j=a-b;
       return L;
       }
```

```
void main()
{
   struct twoints A;

   A = fun(4,3);           //wstawiamy strukturę zwracaną przez funkcję fun do A

   printf("suma = %d\n roznica =%d\n ", A.i, A.j);

   system("pause");
 }
```

Ćwiczenie 3.1.

Zmodyfikuj program z ćwiczenia 2.2. Zmodyfikuj funkcję double gestosc(struct probka X), tak aby zwracała strukurę zawierającą dwie wartości – objętość i gęstość próbki (utwórz nową strukturę, np. struct wyniki). Wykorzystaj ją w funkcji main() do wydrukowania objętości i gęstości próbek.

/ mgr inż. Karol Pietrak, Zakład Termodynamiki, ITC PW. Wykorzystano źródła - Wikipedia