Temat: Struktury danych w języku C++

Struktury:

- **Struktura danych** - sposób przechowywania danych w pamięci komputera.
  - Zmienne w strukturze nazywamy polami, a funkcje na nich działające nazywamy metodami.
  - Struktura stanowi wzór, na podstawie którego powołane są do życia konkretne jej egzemplarze zwane obiektami.

Jak używać struktur:

- Konstrukcja najprostszej struktury:
  - struct nazwa {};

```
1   #include <iostream>
2   #include <string>
3
4   using namespace std;
5
6   struct osoba {
7       string imie;
8       string nazwisko;
9       int wiek;
10  };
11
12  int main(){
13      osoba s1 = {"Piotr","Nowak",21};
14      osoba s2;
15      s2.imie = "Katarzyna";
16      s2.nazwisko = "Kowalska";
17      s2.wiek = 20;
18      cout<<"Imie: "<<s1.imie<<", Nazwisko: "<<s1.nazwisko<<", Wiek: "<<s1.wiek<<endl;
19      cout<<"Imie: "<<s2.imie<<", Nazwisko: "<<s2.nazwisko<<", Wiek: "<<s2.wiek<<endl;
20      return 0;
21  }
22
```

pola struktury

```
Imie: Piotr, Nazwisko: Nowak, Wiek: 21
Imie: Katarzyna, Nazwisko: Kowalska, Wiek: 20

Process returned 0 (0x0)   execution time : 0.935 s
Press any key to continue.
```

Konstruktor i destruktor:

- **Konstruktor** – to „funkcja”, która uruchamia się podczas tworzenia nowego obiektu struktury
  - Nazwa () - funkcja o nazwie struktury bez typu i nie posiada return

- **Destruktor** – to „funkcja”, która uruchamia się podczas usuwania obiektu struktury
  - ~Nazwa () – definicję rozpoczynajmy od znaku „~”

```
1  #include <iostream>
2  #include <string>
3
4  using namespace std;
5
6  struct Osoba
7  {
8      string imie;
9      string nazwisko;
10     int wiek;
11
12     Osoba(string a, string b, int c)
13     {
14         imie = a;
15         nazwisko = b;
16         wiek = c;
17         cout<<"Konstruktor 1"<<endl;
18     };
19
20     Osoba()
21     {
22         imie = "";
23         nazwisko = "";
24         wiek = 0;
25         cout<<"Kostruktor 2"<<endl;
26     };
27     ~Osoba()
28     {
29         cout<<"Destruktor"<<endl;
30     };
31 };
32
33 int main()
34 {
35     Osoba s1 ("Piotr","Nowak",21);
36     Osoba s2;
37     cout<<"Imie: "<<s1.imie<<", Nazwisko: "<<s1.nazwisko<<", Wiek: "<<s1.wiek<<endl;
38     return 0;
39 }
```

Konstruktor

Konstruktor bezargumentowy

Destruktor

```
Konstruktor 1
Kostruktor 2
Imie: Piotr, Nazwisko: Nowak, Wiek: 21
Destruktor
Destruktor

Process returned 0 (0x0)   execution time : 0.893 s
Press any key to continue.
```

Metody:

- **Sposoby różnią się miejscem implementacji metod:**
  - Implementacja razem z definicją struktury.
  - Rozdzielenie implementacji i definicji struktury.

```
#include <iostream>
#include <string>

using namespace std;

struct Osoba
{
    string imie;
    string nazwisko;
    int wiek;

    void urodziny(){
        wiek = wiek + 1;
    }
    void slub(string a);
};

void Osoba::slub(string a){
        nazwisko = a;
    }

int main()
{
    Osoba s1 ={"Joanna","Nowak",21};
    Osoba s2;
    cout<<"Imie: "<<s1.imie<<", Nazwisko: "<<s1.nazwisko<<", Wiek: "<<s1.wiek<<endl;
    s1.urodziny();
    cout<<"Imie: "<<s1.imie<<", Nazwisko: "<<s1.nazwisko<<", Wiek: "<<s1.wiek<<endl;
    s1.slub("Kowalska");
    cout<<"Imie: "<<s1.imie<<", Nazwisko: "<<s1.nazwisko<<", Wiek: "<<s1.wiek<<endl;
    return 0;
}
```

```
Imie: Joanna, Nazwisko: Nowak, Wiek: 21
Imie: Joanna, Nazwisko: Nowak, Wiek: 22
Imie: Joanna, Nazwisko: Kowalska, Wiek: 22

Process returned 0 (0x0)   execution time : 0.019 s
Press any key to continue.
```

Zadania przykładowe:

1. Zapamiętaj 10 współrzędnych punktów na płaszczyźnie. Użyj tablicy struktur. Współrzędne punktów pobierz od użytkownika, mają być one liczbami całkowitymi. Wszystkie pobrane punkty w sposób czytelny wypisz na ekranie.

```
#include <iostream>
#include <string>

using namespace std;

struct punkt
{
    int x,y;
};

int main()
{
    int N=10;
    punkt p[N];
    for(int i=0;i<N;i++){
        cout<<"Podaj x"<<i<<"=";
        cin>>p[i].x;
        cout<<"Podaj y"<<i<<"=";
        cin>>p[i].y;
    }
    for(int i=0;i<N;i++)
        cout<<"P"<<i+1<<"=("<<p[i].x<<","<<p[i].y<<
        ") ";
    return 0;
}
```

```
Podaj x0=1
Podaj y0=2
Podaj x1=3
Podaj y1=4
Podaj x2=5
Podaj y2=6
Podaj x3=7
Podaj y3=8
Podaj x4=9
Podaj y4=5
Podaj x5=3
Podaj y5=2
Podaj x6=56
Podaj y6=7
Podaj x7=8
Podaj y7=4
Podaj x8=2
Podaj y8=7
Podaj x9=8
Podaj y9=4
P1=(1,2) P2=(3,4) P3=(5,6) P4=(7,8) P5=(9,5) P6=(3,2) P7=(56,7) P8=(8,4) P9=(2,7) P10=(8,4)
Process returned 0 (0x0)   execution time : 8.918 s
Press any key to continue.
```

2. Dana jest struktura **struct punkt2d {float x,y;};**
   Napisać funkcję o nagłówku **punkt2d srodekOdcinka(Punkt2d p1, Punkt2d p2)**, która będzie zwracała współrzędne środka odcinka dla zadanych jej końców przez parametry jej wywołania. Pobierz od użytkownika współrzędne obu końców odcinka i użyj funkcji do obliczenia współrzędnych jego środka. Wyświetl czytelnie wyniki na ekranie.

```
1  #include <iostream>
2  #include <string>
3  using namespace std;
4
5  struct punkt2d
6  {
7      float x,y;
8  };
9
10 punkt2d srodekOdcinka (punkt2d p1, punkt2d p2)
11 {
12     punkt2d sr;
13     sr.x = (p1.x + p2.x)/2;
14     sr.y = (p1.y + p2.y)/2;
15     return sr;
16 }
17
18 void wypiszP(string nazwa, punkt2d p)
19 {
20     cout <<"a"<<"=("<<p.x<<","<<p.y<<")"<<endl;
21 }
22
23 int main()
24 {
25     punkt2d p1, p2, psr;
26     cout<<"Podaj x1="; cin>>p1.x;
27     cout<<"Podaj y1="; cin>>p1.y;
28     cout<<"Podaj x2="; cin>>p2.x;
29     cout<<"Podaj y2="; cin>>p2.y;
30     cout<<endl;
31
32     psr=srodekOdcinka(p1,p2);
33     wypiszP("P1",p1);
34     wypiszP("P2",p2);
35     wypiszP("Psr",psr);
36     return 0;
37
38 }
```

Zadania do samodzielnej realizacji:

1. (2p) Zdefiniuj strukturę o nazwie Auto zawierającą:
    a. **pole bak typu double**,
    b. **konstruktor Auto()** inicjujący zerem wartość pola bak,
    c. metodę **void tankowanie(double ilosc)** umożliwiającą zatankowanie samochodu, przyjmijmy, że do baku nie można zatankować więcej niż 50 litrów paliwa,
    d. metodę **void jazda(double km)** symulującą zużycie paliwa podczas jazdy samochodem: - przyjmijmy, że 1 litr paliwa umożliwia przejechanie 10 km, - podczas jazdy samochodem może braknąć paliwa, należy wtedy to zasygnalizować,
    e. metodę publiczną **void stan_paliwa()** wyświetlającą ilość paliwa w baku.

    Przetestuj działanie metod struktury Auto.

2. (3p) Zdefiniuj strukturę o nazwie **Wielomian** zawierającą liczbę będącą stopniem wielomianu (maksymalnie 100) i tablicę z rzeczywistymi współczynnikami tego wielomianu. Zdefiniuj funkcje operujące na wielomianach:
    a. **Wielomian suma(Wielomian w1, Wielomian w2)**,
    b. **Wielomian roznica(Wielomian w1, Wielomian w2)**,
    c. **Wielomian iloczyn(Wielomian w1, Wielomian w2)**,
    d. **void wypisz(Wielomian w)**.

    Przetestuj działanie utworzonych funkcji. Wyniki w sposób czytelny wyświetl na ekranie.