Rozdział 7. Struktury, tablice struktur w języku C/C++

W języku C++, jak w każdym języku obiektowym, mamy możliwość definiowania własnych typów danych, wraz z określeniem operacji, jakie na tych danych można wykonywać. Służą do tego (jak w Javie) klasy, występujące w C++, z powodów głównie historycznych, pod dwoma nazwami: klas i struktur.

Jednak typy złożone istnieją również w czystym C. Mają one postać unii i struktur. W C++ implementacja struktur została rozszerzona (i jest w zasadzie taka jak dla klas), ale warto zdawać sobie sprawę, jak wyglądają struktury zgodne z implementacją w C. Tego typu struktury, które będziemy tworzyć w dalszej części niniejszego rozdziału, są bardzo często używane nawet w programach napisanych w C++, głównie ze względu na to, że C-strukturami są często typy zdefiniowane w standardowych bibliotekach.

7.1 Zasady tworzenia struktur

Dlaczego jest nam potrzebne definiowanie nowych typów danych? Otóż w pewnych sytuacjach, typy zmiennych, które oferuje język C++ mogą być niewystarczające. Mamy do dyspozycji szereg typów prostych takich jak **int, char, bool**czy typy złożone zwane **tablicami**. Tablica to taka struktura danych, która przechowuje wiele elementów **tego samego typu.** Co jeśli potrzebujemy pracować na obiektach, których składowe są różnych typów? Na ratunek przychodzą nam **struktury,** czyli złożone typy danych, które definiuje programista. Załóżmy, że chcemy przechować dane osobowe takie jak:

* Imię
* Nazwisko
* Rok urodzenia

Moglibyśmy stworzyć cztery zmienne oraz dobrać im typy lub stworzyć nowy typ, którego elementami będą omawiane składowe. Taki typ nazywamy **strukturą.**

Definicja Struktury może mieć następującą postać:

**struct** Nazwa {

Typ1 skladnik1;

Typ2 skladnik2;

...

};

Nazwy Typ1 i  Typ2 są tu nazwami typów. Nazwy skladnik1 i  skladnik2 dowolnymi identyfikatorami pól tej struktury. Każda zmienna tego typu strukturalnego będzie zawierać składowe o typach i nazwach odpowiadających polom struktury. Na razie jednak jest to tylko definicja typu: nie istnieją jeszcze żadne tego typu zmienne. Średnik na końcu, po nawiasie zamykającym, jest konieczny!   
W C++ po takiej samej definicji nazwą typu jest po prostu Nazwa. Natomiast w C nazwą tego typu jest struct Nazwa - konieczne jest zatem powtórzenie słowa kluczowego struct przy późniejszym użyciu nazwy tego typu.

Kiedy już zdefiniowaliśmy typ, możemy definiować zmienne tego typu; składnia jest dokładnie taka sama jak składnia deklaracji/definicji zmiennych typów wbudowanych (jak **double** czy int):

Nazwa c, d;

Można też definiować zmienne tego typu bezpośrednio za definicją, a przed średnikiem (do tego właśnie służy średnik, o którym wiecznie się zapomina):

**struct** Nazwa {

Typ1 skladnik1;

Typ2 skladnik2;

} a, b, c, d;

definiuje typ strukturalny i od razu definiuje cztery zmienne tego typu. Ten ostatni zapis daje możliwość tworzenia obiektów struktur anonimowych:

**struct** {

Typ1 skladowa1;

Typ2 skladowa2;

} a, b;

Za pomocą takiej konstrukcji stworzyliśmy dwa obiekty strukturalne o nazwach **a** i  **b**. Każdy z nich zawiera dwie składowe odpowiadające dwóm polom w definicji struktury. Do składowych tych można się odnosić w sposób opisany poniżej, jak do składowych zwykłych struktur posiadających nazwę. Nie da się już jednak zdefiniować innych obiektów tego samego typu, bo typ ten nie ma żadnej nazwy i nie ma jak się do niego odwołać!

Każdy pojedynczy obiekt typu zdefiniowanego jako Struktura zawiera tyle składowych i takich typów, jak to zostało określone w definicji tej Struktury. Jeśli zdefiniowany (utworzony) jest obiekt, to do jego składowych odnosimy się za pomocą operatora wyboru składowej. Operator ten ma dwie formy, w zależności od tego, czy odnosimy się do obiektu poprzez jego nazwę, czy poprzez nazwę wskaźnika do niego. Jeśli **a** jest nazwą *obiektu*, to do jego składowej o nazwie skladnik odnosimy się za pomocą operatora „kropki”:

a.skladnik

Ta sama reguła stosowałaby się, gdyby **a** było nazwą *odnośnika* do obiektu - pamiętajmy jednak, że w czystym C odnośników (referencji) nie ma. Natomiast jeśli ***pa*** jest *wskaźnikiem* do pewnego obiektu struktury, to do tej samej składowej odnosimy się za pomocą operatora „strzałki”:

pa->skladnik

Tworząc obiekt typu C-struktury można go od razu zainicjować. Składnia jest podobna do tej, jakiej używamy do inicjowania tablicy:

**struct** Nazwa ob = {wyrazenie\_1, wyrazenie\_2};

gdzie wyrazenie\_1 i  wyrazenie\_2 są wyrażeniami, których wartości mają posłużyć do zainicjowania składowych obiektu w kolejności takiej, w jakiej zadeklarowane zostały odpowiednie pola struktury. Inicjatorów może być mniej niż pól; składowe odpowiadające pozostałym polom zostaną wtedy zainicjowane zerami odpowiedniego typu. Słowo kluczowe **struct** w powyższej instrukcji może być pominięte w C++, natomiast jest niezbędne w C.

7.2 Zasady tworzenia Uni

Unia jest typem definiowanym przez użytkownika, w którym można przechowywać różne typy danych, ale w danym momencie programu tylko jeden. W unii wszystkie składowe obiektu umieszczane są pod tym samym adresem. Zatem w każdej chwili dostępna jest tylko jedna składowa. Zauważmy bowiem, że wpisanie którejś ze składowych zamazuje poprzednią, bo była ona umieszczona w dokładnie tym samym miejscu w pamięci komputera. Wynika z tego, że rozmiar obiektu unii musi być taki, aby mieściła się w nim składowa o największym rozmiarze, ale nie musi być większy, choć może - zależy to od typów i rozmiarów poszczególnych składowych.

Unię deklaruje się podobnie jak strukturę:

**union** nazwa

{

tu definicje alternatywnych pól;

};

Deklaracje zmiennej oraz inicjalizacja wartości odbywa się tak samo jak w przypadku struktur. Należy pamiętać, że po wczytaniu wartości pod kolejne pole poprzednia wartość zostaje zamazana. Zaletą unii jest fakt, że można w niej zapisywać kolejno wartości różnego typu nie zajmując kolejnych obszarów pamięci, ale wadą jest to, że w jednym czasie możemy przechowywać tylko jeden typ wartości

**7.3 Pola bitowe**

Oprócz zwykłych pól w strukturach możemy zastosować tak zwane pola bitowe. Pole bitowe to wydzielenie pewnej stałej porcji bitów na daną zmienną. Na przykład zmienna typu int zajmuje w pamięci 4 bajty = 4\*8 bitów=32 bity. Możemy ją okroić lub rozszerzyć o porcję bitów dostosowaną do potrzeb programu. Pamięć jaką będzie zajmować takie pole będzie zawsze wielokrotnością bajtów danego typu zmiennej. Np. jeśli stworzymy pole typu int na 4 bity, to i tak zostanie przydzielona pamięć na cały typ int czyli 4 bajty = 32 bity, a używać będziemy mogli tylko tych czterech bitów. Na czym więc polega tu oszczędność? Gdy stworzymy na przykład cztery pola bitowe typu int, i każde będzie zajmowało 6 bitów, to w pamięci zostanie zarezerwowany obszar na jeden int, czyli 32 bity, ponieważ 4\*6 = 24≤ 32, i właśnie tu mamy oszczędność pamięci.

Można deklarować pola bitowe tylko typu int: ze znakiem (signed) lub bez znaku (unsigned) - podobnie jak dla normalnych typów całkowitych. Liczbę bitów przeznaczonych dla danego pola deklarujemy po dwukropku

**typ** nazwa : [ilość bitów];

Przykładowo jeśli zadeklarujemy :

struct wersja

{

int x :3;

int y :7;

}

To zmienna pole x może przechowywać liczby z zakresu [0, 7], bo istnieje osiem zero-jedynkowych kombinacji trzech bitów:

7 6 5 4 3 2 1 0

111 110 101 100 011 010 001 000

Z kolei pole y może przechowywać liczby z zakresu [0, 127] gdyż liczba binarna 1111111 odpowiada liczbie 127 w systemie dziesiętnym. Gdybyśmy nie ograniczyli rozmiaru pól x i y to każde z nich zajmowałoby 32 bity. Natomiast dzięki zastosowaniu ograniczenia ilości bitów o ograniczonym rozmiarze kompilator „upakuje” zmienne na obszarze 32 bitów, gdyż 3+7=10 bitów<32 bity.

7.4 Przykładowe zadania

***Zadanie 7.1***

*Załóżmy więc, że chcemy zdefiniować strukturę o nazwie osoba, zawierającą trzy pola: imię, nazwisko, rok\_urodzenia. Następnie chcemy wczytać dane z klawiatury pod zmienną tego typu.*

Kod definicji struktury będzie następujący:

struct osoba{

char imie[20];

char nazwisko[30];

int rok\_urodzenia;

};

Zadeklarowanie zmiennej a typu osoba:

osoba a;

Wczytanie wszystkich wartości zmiennej a typu osoba:

cout<<"imie=";cin>>a.imie;

cout<<"nazwisko=";cin>>a.nazwisko;

cout<<"rok urodzenia=";cin>>a. rok\_urodzenia;

Pełny kod programu prezentuje Listing 7.1.

1. #include <iostream>
2. using namespace std;
3. struct osoba{
4. char imie[20];
5. char nazwisko[30];
6. int rok\_urodzenia;
7. };
8. int main()
9. {
10. osoba a;
11. cout<<"Podaj imie=";cin>>a.imie;
12. cout<<"Podaj nazwisko=";cin>>a.nazwisko;
13. cout<<"Podaj rok urodzenia=";cin>>a. rok\_urodzenia;
15. cout<<"wczytane dane:\n";
16. cout<<"imie="; cout<<a.imie<<endl;
17. cout<<"nazwisko="; cout<<a.nazwisko<<endl;
18. cout<<"rok urodzenia="; cout<<a.rok\_urodzenia<<endl;
19. system ("pause");
20. }

**Listing 7.1.** Kod programu uwzględniający rozwiązanie Zadania 7.1

Zwróćmy uwagę, że definicja struktury umieszczona jest przed funkcją main(). Gwarantuje to, że stworzony typ strukturalny będzie znany globalnie w każdym miejscu programu.

***Zadanie 7.2***

*Utwórz bazę danych osób wczytanych za pomocą struktury z zadania 7.1.*

W sposób który został przedstawiony w rozwiązaniach zadania 7.1 widzimy, że możemy wczytać dane tylko jednej osoby. Jak natomiast przechować bazę takich osób? Najprostszym rozwiązaniem jest zastosowanie tablicy, której elementami będą obiekty typu zadeklarowanej osoby.

I tak oto mając zadeklarowaną strukturę osoba z zadania 7.1 definicja tablicy przechowującej 1000 zmiennych tego typu będzie następująca:

Osoba baza[1000];

Kod realizujący zadanie 7.2 zgodnie z podejściem zaprezentowanym na z Listingu 7.1 przedstawia Listing 7.2.

1. #include <iostream>
2. using namespace std;
3. struct osoba{
4. char imie[20];
5. char nazwisko[30];
6. int rok\_urodzenia;
7. };
8. int main()
9. {
10. osoba baza[1000];
11. cout<<"ile osob chcesz wprowadzic do bazy?\n";
12. int n;
13. cin>>n;
14. for (int i=0; i<n; i++)
15. {
16. cout<<"podaj dane"<<i+1<<"-tej osoby\n";
17. cout<<"imie=";cin>>baza[i].imie;
18. cout<<"nazwisko=";cin>>baza[i].nazwisko;
19. cout<<"rok urodzenia=";cin>>baza[i].rok\_urodzenia;
20. }
21. for (int i=0; i<n; i++)
22. {
23. cout<<"Dane"<<i+1<<"-tej osoby\n";
24. cout<<"Wczytane imie=";cout<<baza[i].imie<<endl;
25. cout<<"Wczytane nazwisko=";cout<<baza[i].nazwisko<<endl;
26. cout<<"Wczytany rok urodzenia=";cout<<baza[i].rok\_urodzenia<<endl;
27. cout<<endl;
28. }
29. system ("pause");
30. }

**Listing 7.2.** Kod programu uwzględniający Rozwiązanie 1 Zadania 7.2

***Zadanie 7.3***

*Zmodyfikuj zadanie 7.1 w taki sposób, że zamiast pola rok\_urodzenia struktura będzie zawierała dokładna datę tj. rok, miesiąc i dzień.*

Możemy oczywiście zamiast zmiennej rok\_urodzenia dodać trzy zmienne: rok, miesiąc, dzień. Nie będzie to jednak optymalne rozwiązanie. Zdecydowanie lepszym będzie zdefiniowanie dodatkowej struktury Data i użycie jest w strukturze Osoba. To co jest istotne, to że definicja struktury Data musi pojawić się w kodzie programu przed definicją struktury Osoba**.**

Kod programu realizującego to zadanie może mieć następująca postać:

1. #include <iostream>
2. using namespace std;
3. struct Data{
4. int rok, miesiac, dzien;
5. };
6. struct osoba{
7. char imie[20];
8. char nazwisko[30];
9. Data data\_urodzenia;
10. };
11. int main()
12. {
13. osoba a;
14. cout<<"Podaj imie=";cin>>a.imie;
15. cout<<"Podaj nazwisko=";cin>>a.nazwisko;
16. cout << "Podaj date urodzenia:\n";
17. cout<<"rok:\n";
18. cin >> a.data\_urodzenia.rok;
19. cout<<"miesiac:\n";
20. cin >>a.data\_urodzenia.miesiac;
21. cout<<"dzien:\n";
22. cin >> a.data\_urodzenia.dzien;
24. cout<<"wczytane dane:\n";
25. cout<<"imie="; cout<<a.imie<<endl;
26. cout<<"nazwisko="; cout<<a.nazwisko<<endl;
27. cout << "Data urodzenia:";
28. cout << a.data\_urodzenia.rok<<"-"<< a.data\_urodzenia.miesiac<<"-"  
     << a.data\_urodzenia.dzien;
29. system ("pause");
30. }

**Listing 7.3** Rozwiązanie Zadania 7.3

***Zadanie 7.4***

*Napisz program, w którym zadeklarujesz unie mogącą przechowywać zmienne typu rzeczywistego lub znakowego. Wczytaj i wypisz wartości tych zmiennych.*

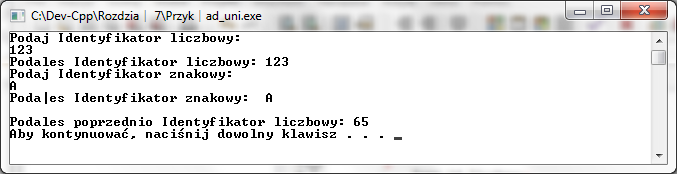
Realizacja zadania może mieć następującą postać:

1. #include <iostream>
2. using namespace std;
3. union id
4. {
5. long id\_liczbowy;
6. char id\_lancuchowy[20];
7. };
8. int main()
9. {
10. id przedmiot;
11. cout << "Podaj Identyfikator liczbowy: \n" ;
12. cin>> przedmiot.id\_liczbowy;
13. cout<<"Podales Identyfikator liczbowy: "<<przedmiot.id\_liczbowy<<endl;
15. cout << "Podaj Identyfikator znakowy: \n" ;
16. cin>> przedmiot.id\_lancuchowy;
17. cout<<"Podałes Identyfikator znakowy: "  
     <<przedmiot.id\_lancuchowy<<endl<<endl;
18. cout<<"Podales poprzednio Identyfikator liczbowy: "  
     <<przedmiot.id\_liczbowy<<endl;
19. system ("pause");
20. }

**Listing 7.4.** Przykładowy program wykorzystujący unię.

Zauważmy, że w linii 13 wczytujemy najpierw do unii zmienną typu liczbowego, następnie w linii 16 zmienną typu znakowego. Co zobaczymy na ekranie po wydaniu polecenia z linii 19 ponownego wypisania wartości liczbowej? Dlaczego?

Skoro w uni można przechowywać tylko jedną zmienna wartość liczbową po wykonani kodu z linii 17 wartość liczbowa zostanie zamazana i nie będzie już do niej dostępu. Odpowiedź na pytanie co zostało wypisane zamiast pozostawiamy Studentom. Przykładowe uruchomienie programu przedstawia Rysunek 7.1.



**Rysunek 7.1.** Przykładowy wynik uruchomienia programu z Listingu 7.7

7.5 Zadania

1. Napisz program przechowujący bazę zawierającą dane Studentów zawierająca: Nazwisko, kierunek, listę przedmiotów i oceny z nich. Napisz program, który będzie:
2. Dodawał studenta do listy;
3. Wyszukiwał studentów średniej z semestru większej od zadanej jako parametr, i wypisywał ich nazwiska
4. Napisz program definiujący typ strukturalny TSamochod (marka, silnik, rocznik, numer nadwozia) oraz bazę danych przechowującą zmienne typu TSamochod. Napisz program, który będzie dodawał samochody do bazy, a następnie w zależności od wyboru użytkownika: :
   1. Wypisywał z bazy samochody danej marki.
   2. Wypisywał z bazy samochody z danego rocznika.
   3. Wyszukiwał, czy w bazie jest samochód o danym numerze nadwozia a następnie wypisywał jego dane na ekran.
5. Napisz program definiujący typ strukturalny oraz 100-elementową tablicę pozwalającą przechowywać informacje o książkach znajdujących się w bibliotece (tytuł, autor, indeks, rok wydania, cena) . Napisz program, który będzie dodawał książki do bazy a następnie:
   1. Wypisywał książki wydane w roku zadanym jako parametr
   2. Wypisywał książki , których cena jest większa od zadanej jako parametr
   3. Wypisywał dane książki danego autora
6. Zdefiniuj strukturę przechowującą dane katalogowe książki w bibliotece (tytuł = 50 znaków;   
   autor = 30 znaków; liczba stron = liczba całkowita dodatnia; wypożyczona = zmienna logiczna) oraz 100-elementową tablicę takich struktur. Napisz program, który będzie dodawał książki do bazy,  
   a następnie:
   1. Wypisywał książki o ilości stron większej niż zadana jako parametr,
   2. Wypisywał książki danego autora
   3. Wypisywał listę książek niewypożyczonych.
7. Zdefiniuj typ strukturalny oraz 50-cio elementową tablicę struktur pozwalającą przechowywać informacje o towarach w sklepie RTV (nazwa\_towaru, ilosc\_sztuk, cena, waga). Napisz program, który będzie dodawał towary do bazy, a następnie:
   1. Wypisywał listę towarów, których jest w sklepie mniej niż liczba zadana jako parametr
   2. Wypisywał ilość sztuk towaru o nazwie zadanej jako parametr
   3. Wypisze dane najtańszego towaru w sklepie.
8. Dane są dwie zmienne rekordowe o strukturze: nazwa, adres, kwartal[4]. Napisz program, który wczyta dane do obydwu zmiennych i wypisze 1, gdy rekordy mają takie same dane i 0 w przeciwnym razie.
9. Dana jest n-elementowa tablica ze strukturami opisującymi współrzędne punktów na płaszczyźnie PUNKT( float x, float y, char cwiartka[5] ). Napisz program, który sprawdzi w której ćwiartce układu współrzędnych znajduje się dany punkt, i tę informację zapisze do tablicy w pole ćwiartka (I,II,III,IV)(ESz) Napisz program umożliwiający przechowywanie danych o strukturze: nazwisko, adres, pensja\_brutto, czy\_pali. Strukturę nazwij Pracownik.   
   Wprowadź dane i wypisz je na ekran.   
   Pomoc: część rozwiązania:

{

struct Pracownik

{

char nazwisko[20];

char adres[20];

float brutto;

bool pali;

};

Pracownik a;

cout<<"nazwisko=";cin>>a.nazwisko;

cout<<"adres=";cin>>a.adres;

cout<<"brutto=";cin>>a.brutto;

cout<<"czy\_pali(0/1)";

cin>>a.pali;

}

1. Dana jest lista osób o strukturze z poprzedniego zadania zapisana w tablicy. Wprowadź dane n osób do tablicy a następnie napisz program który:

* policzy sumę pensji brutto wszystkich pracowników,
* Wypisze wartość maksymalnej pensji brutto,
* Wypisze wartość minimalnej pensji brutto,
* Wypisze nazwiska i imiona osób palących.

1. Dana jest lista osób o strukturze z zadania 1 zapisana w tablicy. Napisz program, który przepisze wszystkie osoby mające wyższe brutto niż średnia z brutto do nowej tablicy. Wypisz na ekran dane z tablicy początkowej i wynikowej.
2. Napisz program przechowujący bazę danych osób z firmy XXX. Baza powinna zawierać informację: imie, nazwisko, rok urodzenia, adres. Napisz program, który wczytuje dane do bazy oraz wypisuje osoby urodzone po danym roku. Zdefiniuj strukturę do przechowywania daty.
3. Napisz program, w którym zdefiniujesz unię mogącą przechowywać zmienna typu całkowitego, rzeczywistego lub tekstowego. Wczytaj zmienne różnych typów i przetestuj na różnych etapach programu co zostało wypisane i dlaczego.
4. Dany jest zbiór polskich miast wraz ze współrzędnymi podanymi w układzie 0XY, gdzie x oznacza odległość w kilometrach od Warszawy w kierunku osi X a y odległość w kilometrach od Warszawy w kierunku osi Y. Napisz program, który wypisze miasta najbardziej odległe od siebie. Nie uwzględniamy efektów związanych z krzywizną ziemi. (MB)
5. Należy zdefiniować N-elementową tablice struktur opisujących płyty DVD. Każda płyta powinna być reprezentowana przez tytuł, dane (struktura z polami: gatunek, wersja, czas trwania ) oraz cenę. Tablice należy zainicjować kilkoma swoimi ulubionymi filmami. Napisać program, który wyliczy średnią cenę płyt danego gatunku znajdujących się w tablicy a następnie wypisze tytuły tych filmów, które mają cenę poniżej średniej.
6. Zdefiniuj strukturę o nazwie LiczbaZespolona, zawierającą dwa pola typu double: a, b. Następnie zaimplementuj kod pozwalający w zależności wyboru użytkownika::
   1. zsumować dwie liczby zespolone,
   2. wyznaczyć moduł liczby zespolonej,
   3. wyznaczyć argument (kąt fi) dla liczby zespolonej,
   4. znaleźć pierwiastki n-tego stopnia dla zadanej liczby zespolonej z – wykorzystaj postać
   5. Eulera.
7. Zdefiniuj typ strukturalny oraz 100-elementową tablicę pozwalającą przechowywać informacje o samochodach w auto-komisie (marka, przebieg, kolor, cena). Napisz program, który będzie dodawał samochody do bazy, a następnie w zależności od wyboru użytkownika:
   1. Wypisywał dane samochodu o najmniejszym przebiegu;
   2. Wypisywał dane samochodu o największym przebiegu,
   3. Wypisywał dane samochodów o cenie niższej niż zadana jako parametr.
8. Zdefiniuj typ struktury przechowującej dane personalne pracownika (nazwisko - 20 znaków; imię - 15 znaków; pensja - liczba rzeczywista; płeć - znak 'm' lub znak 'k') oraz 100-elementową tablicę takich struktur. Napisz program, który będzie dodawał pracowników do bazy, a następnie wypisywał czy w danej firmie większa jest średnia pensja kobiet czy mężczyzn.
9. Zdefiniuj typ struktury przechowującej dane personalne studenta (nazwisko; imię; stypendium = ilość złotych i groszy; zaliczenie = zmienna logiczna) oraz 100-elementowa tablicę takich struktur. Napisz program, który będzie dodawał studentów do bazy, a następnie w zależności od wyboru użytkownika:
   1. Wypisywał dane studenta o najniższym stypendium;
   2. Wypisywał dane studenta o najwyższym stypendium;
   3. Wypisywał dane studenta o najniższym stypendium, który zaliczył semestr;
   4. Wypisywał dane studenta o najwyższym stypendium, który nie zaliczył semestru;
10. Zdefiniuj typ struktury przechowującej dane personalne studenta (nazwisko : 25 znaków; imię : 15 znaków; wiek : liczba o wartościach 0-120; zaliczenie : zmienna logiczna) oraz 50-elementowa tablicę takich struktur. Napisz program, który będzie dodawał studentów do bazy, a następnie w zależności od wyboru użytkownika:
    1. Wypisywał dane najmłodszego studenta,
    2. Wypisywał dane wszystkich studentów, których nazwiska i imiona zaczynają się na tą samą literę (np. **K**asia **K**owalska, **N**atalia **N**owak ),
    3. Wypisywał dane studentów o nazwiskach zaczynających się na literę zadaną jako parametr.
11. Zdefiniuj typ struktury przechowującej dane o pociągach w rozkładzie jazdy (Miasto docelowe: 30 znaków; długość trasy : liczba całkowita bez znaku; godzina odjazdu: liczba rzeczywista; rodzaj pociągu: znak 'o' - osobowy, 'p' - pospieszny, 'e' - ekspres) oraz 100-elementową tablicę takich struktur. Napisz program, który będzie dane pociągów do bazy, a następnie w zależności od wyboru użytkownika:
    1. Wypisywał dane pociągów odjeżdżających po zadanej jako parametr godzinie
    2. Wypisywał dane pociągów pośpiesznych jadących do zadanego jako parametr miasta,
    3. Wypisywał dane pociągów o najdłuższej trasie.
12. Zdefiniuj strukturę punkt służącą do przechowywania współrzędnych punktów w trójwymiarowej przestrzeni kartezjańskiej. Napisz program, który będzie dodawał punkty do bazy, a następnie w zależności od wyboru użytkownika:
    1. Zwracał najmniejszą spośród odległości pomiędzy dwoma punktami przechowywanymi w bazie wraz z danymi punktów,
    2. Zwracał ilość punktów znajdujących się w I ćwiartce układu
13. Zdefiniuj strukturę punkt4 służącą do przechowywania współrzędnych punktów w czterowymiarowej przestrzeni kartezjańskiej. Do przechowywania współrzędnych w poszczególnych wymiarach wykorzystaj tablicę czteroelementową. Napisz program, w oparciu o dwie tablice tab1 i tab2 typu struct punkt4 oraz przepisuje do nowej tablicy tab3 punkty z tab1 i tab2 posiadające takie same dwie współrzędne. Np. w tab1 mamy punkty (3,4,6,1) (2,5,8,1), a w tab2 mamy (2,4,9,1), (3,9,0,1) to do tab3 będzie przepisane (3,4,6,1) oraz (2,4,9,1).
14. Zdefiniuj unię Liczba, która może służyć w zależności od potrzeb do przechowywania liczby wymiernej lub liczby całkowitej. Zdefiniuj strukturę TDane, o dwóch polach polu TP typu int oraz polu Tzawartosc typu TLiczba. Napisz program, który wczytuje zawartość do struktury TDane. Program powinien pytać użytkownika, czy chce wczytać liczbę całkowitą, czy wymierną oraz w zależności od jego wyboru wstawić do pola TP wartość 0 lub 1. Następnie program powinien wczytać do pola Tzawartosc wartość odpowiedniego typu.
15. Zdefiniuj strukturę dane\_pracownikow zawierającą pola: imie,nazwisko, plec, stan\_cywilny. W zależności od płci pole stan\_cywilny powinno moc mieć jedną z dwóch wartości wolny lub zonaty dla mężczyzn i wolna lub mezatka dla kobiet. Napisz program, który będzie dodawał pracowników do bazy, a następnie w zależności od wyboru użytkownika:
    1. Wypisywał dane wszystkich zamężnych kobiet.
    2. Wypisywał dane wszystkich wolnych mężczyzn.
16. Napisz program, który będzie pozwalał badać sprawozdania finansowe przedsiębiorstw za ostatnie cztery kwartały. Dane przedsiębiorstw są pamiętane w tablicy i poszczególne elementy tablicy mają postać: Nazwa, data, Zysk\_za\_kwartal[1..4], nip, uwagi. Napisz program, który będzie dodawał dane przedsiębiorstw do bazy, a następnie wypisz dane przedsiębiorstwa o największej i o najmniejszej wartości średniego zysku.
17. Napisz program w którym utworzysz tablicę struktur, zawierających dane pracowników (nazwisko, stawkę, numer miesiąca, godziny przepracowane w tymże miesiącu). Napisz program, który będzie dodawał dane pracowników do bazy, a następnie w zależności od wyboru użytkownika:
    1. Wypisze dane pracownika, który w całym roku przepracował najwięcej godzin (czyli suma godzin ze wszystkich miesięcy dla danego pracownika)
    2. Wypisze dane pracownika, który dostał najwyższą jednorazową wypłatę i zapisze jego dane do pliku tekstowego.
18. Dana jest baza struktur Pracownik zawierająca: Nazwisko, Imię, pesel, adres (będący strukturą), pensje brutto. Napisz program, który będzie dodawał pracowników do bazy, a następnie wypisywał dane pracowników o pensji brutto większej od zadanej, jako parametr.

### Dana jest baza struktur Student zawierająca: Nazwisko, Imię, pesel, adres (będący strukturą), średnią z każdego semestru studiów (max 7 semestrów). Napisz program, który będzie dodawał studentów do bazy, a następnie wyszukiwał studentów o średniej za dany semestr większej od zadanej, jako parametr.

1. Napisz program, który wykorzystując strukturę przechowuje dane opisujące osoby (wzrost – podany w metrach; imię i nazwisko – string lub łańcuch char; wiek – w latach; waga – w kg). Napisz program, który będzie dodawał osoby do bazy, a następnie porównuje osoby drukując na ekranie zestawienie: która osoba jest wyższa (imię i nazwisko tej osoby), która osoba jest najlżejsza (imię i nazwisko tej osoby) oraz która osoba jest najmłodsza (imię i nazwisko tej osoby).
2. Zdefiniuj strukturę Adres do przechowywania następujących danych: miejscowość, poczta, nr\_domu, nr\_mieszkania, województwo. Wykorzystaj ją do przechowywania danych o mieszkańcach: imie, nazwisko, miejsce\_zamieszkania.   
   Napisz program, który umożliwi wprowadzenie danych o n mieszkańcach (n<=10) i wypisze imiona i nazwisko tych osób, które mieszkają w podanym przez użytkownika województwie.
3. Napisz program, który wypełni tablicę studentów następującymi danymi  
   imie – imie\_nr, gdzie nr to kolejna liczba  
   nazwisko - nazwisko\_nr, gdzie nr to kolejna liczba  
   rok\_studiów – liczba pseudolosowa z przedziału <1,4>  
   kierunek – zależnie od wylosowanej liczby z przedziału <0,3) 0-informatyka, 1-chemia, 2-matematyka  
   data\_urodzenia – pole złożone zawierające dzień, miesiąc i rok. Rok – liczba pseudolosowa z przedziału 1980 – 1997, miesiąc – liczba z przedziału 1-12, dzień – liczba pseudolosowa, zależna od miesiąca i roku.  
   Napisz program, który wypisze imię i nazwisko osoby najmłodszej i najstarszej, jeśli jest kilka takich osób powinno być podane nazwisko pierwszej.
4. Zmodyfikuj zadanie 2, uzupełniając strukturę do przechowywania danych studentów o tablicę 3 ocen – liczby rzeczywiste. Wypisz nazwisko studenta o najwyższej średniej.
5. Zdefiniuj strukturę do przechowywania danych o wektorach na płaszczyźnie:  
   nazwa np. AB  
   współrzędne x,y punktu początkowego  
   współrzędne x,y punktu końcowego  
   Wprowadź dane do tablicy wektorów. Napisz program, który
   1. obliczy długości wektorów ,
   2. poda nazwę wektora o największej długości.,
   3. sprawdzi czy w tablicy są dane o wektorach o tej samej długości
6. Zdefiniuj strukturę, do przechowywania danych o pracownikach: imię, nazwisko, tablica poborów brutto z 12 miesięcy ostatniego roku. Napisz program, który wypisze, w który miesiącu łączne pobory brutto pracowników były najmniejsze.
7. Zamierzasz utworzyć bazę danych do przechowywania książek. Zaproponuj odpowiednią strukturę rekordu dla książki. Do zmiennej typu rekordowego wczytaj do niej dane i wypisz je.
8. Dana jest tablica zawierająca elementy o strukturze: nazwa, kwota. Napisz program, który będzie wczytywał n danych do struktury, tak by dane były posortowane według:
   * 1. nazwy,
     2. kwoty.

Wypisz tablicę wynikową.

1. Dana jest jednowymiarowa n-elementowa tablica punktów 2D. Napisz program, który wypisze 1, gdy nie ma punktów współliniowych i 0, gdy są.
2. Dana jest tablica jednowymiarowa n-elementowa tablica punktów 2D. Napisz program, który wypisze ile trójkątów można zbudować z tych punktów.
3. Dana jest baza danych przechowująca następujące informacje: nazwisko, stanowisko, brutto. Napisz program, który poda następujące statystyki:
   1. średnia z brutto dla całej bazy danych,
   2. średnie z brutto dla poszczególnych stanowisk,

Ponadto program powinien wypisać nazwę tego stanowiska, dla którego występuje największa i najmniejsza średnia.

1. Zdefiniuj typ strukturalny oraz 100 elementową tablicę struktur pozwalającą przechowywać informacje o wejściach na basen (data wejścia, nazwisko pływającego, nazwisko trenera, czas). Data wejścia to struktura typu( dzień, miesiąc, rok). Napisz program, który będzie prowadził statystykę wejść na basen, a następnie:
   * + - * Wypisze wejścia poszczególnych osób
         * Wypisze osoby pływające u trenera podanego przez użytkownika
         * Wypisze ilość wejść w poszczególnych miesiącach
         * Wypisze ogólny czas treningu dla poszczególnych osób
2. Zdefiniuj typ strukturalny oraz 50 elementową tablicę struktur pozwalającą przechowywać przepisy na ciasta( rodzaj, nazwa, czas wykonania, stopień trudności( 1,2,3)). Napisz program, który będzie umożliwiał wprowadzanie danych do książki kucharskiej z ciastami, a następnie:
   * + - * Wypisze ciasta z jednego rodzaju np. torty
         * Obliczy średni czas wykonania ciast
         * Wypisze ciasta z podziałem na stopień trudności.
3. Dana jest n-elementowa tablica ze strukturami opisującymi współrzędne punktów w przestrzeni PUNKT3D ( float x, float y, float z). Napisz program, który sprawdzi czy istnieją w tablicy punkty leżące na tej samej płaszczyźnie. Program powinien wypisać na ekranie informację, które z punktów leżą na tej samej płaszczyźnie.
4. Zdefiniuj typ strukturalny oraz 100 elementową tablicę struktur pozwalającą na przechowywanie informacji o rodzajach zwierząt jakie posiada pewien Farmer. Zwierzęta to krowa, koza, kura, pies i świnia. Cechy tych zwierząt to nazwa, waga, wiek(podany w tygodniach), oraz data nabycia(zakupu). Farmer dostał krowę i świnię kupił 24-04-08r od znajomego, pies przybłąkał się 6.05.02r, natomiast kozę i kurę dostał w prezencie od wójta 23.02.09. Niestety resztę cech musisz określić sam, na podstawie przyjętych średnich dla tych zwierząt. Rezultatem programu ma być:
   * + - * wyświetlenie wszystkich danych w przejrzysty i zrozumiały dla użytkownika sposób, użytkownik nic nie podaje, program wyświetla tylko dane.
         * program ma poprosić o wprowadzenie nazwy i wagi(w kilogramach) zwierzęcia(mają być wprowadzone trzy zwierzaki) , oraz wyświetleniu danych w odwrotny sposób do tego w jaki zostały wprowadzone.
5. Dana jest n-elementowa tablica ze strukturami opisującymi współrzędne punktów na płaszczyźnie (punkt x, punkt y, punkt z ). Typ punkt też jest strukturą postaci   
   ( int pierwsza\_wsp, int druga\_wsp). Napisz program, który sprawdzi czy podane punkty tworzą trójkąty a następnie sprawdzi czy punkt, którego współrzędne wprowadzi użytkownik leży wewnątrz tego trójkąta.

**Zadania ze wstępu do programowania**

1. Napisz program przechowujący bazę zawierającą dane Studentów zawierająca: Nazwisko, kierunek, listę przedmiotów i oceny z nich. Napisz program, który będzie:

* Dodawał studenta do listy;
* Wyszukiwał studentów średniej z semestru większej od zadanej jako parametr, i wypisywał ich nazwiska

1. Napisz program definiujący typ strukturalny oraz 100-elementową tablicę pozwalającą przechowywać informacje o książkach znajdujących się w bibliotece (tytuł, autor, indeks, rok wydania, cena) . Napisz program, który będzie dodawał książki do bazy a następnie:

* Wypisywał książki wydane w roku zadanym jako parametr
* Wypisywał książki , których cena jest większa od zadanej jako parametr
* Wypisywał dane książki danego autora.

1. Zdefiniuj strukturę punkt służącą do przechowywania współrzędnych punktów w trójwymiarowej przestrzeni kartezjańskiej. Napisz program, który będzie dodawał punkty do bazy, a następnie w zależności od wyboru użytkownika:

* Zwracał najmniejszą spośród odległości pomiędzy dwoma punktami przechowywanymi w bazie wraz z danymi punktów,
* Zwracał ilość punktów znajdujących się w I ćwiartce układu.

**Tasks from the introduction to programming**

1. Write a program that stores a database containing Student data, containing: Name, field of study, list of subjects and their grades. Write a program that will:

• added a student to the list;

• He searched for students with a semester average greater than the given semester as a parameter, and entered their names.

1. Write a program that defines the structural type and a 100-element array that allows you to store information about the books in the library (title, author, index, year of publication, price). Write a program that will add books to the database and then:

• Checked out books published in the year set as a parameter

• He wrote out books whose price was higher than the one set as a parameter

• Print data from a book by a given author.

1. Define a point structure to store the coordinates of points in three-dimensional Cartesian space. Write a program that will add points to the base, and then, depending on the user's choice:

• Return the smallest distance between two points stored in the database together with the point data,

• Returned the number of points in the first quarter of the system.