Zadanie 5, Języki Programowania
Zadanie polega na napisaniu klasy wektorN, która będzie reprezentowała dowolny wektor o długości N i umożliwiała wykonanie trzech operacji „+”, „-”, „[]” (dostęp do podanego elementu wektora).

Rozmiar wektora (N) jest podawany w konstruktorze. Domyślna wartość to **N = 4**. Klasa musi przechowywać współrzędne wektora jako zmienne **float** w tablicy. Tablica ma być tworzona dynamicznie w konstruktorze z wykorzystaniem operatora **new**. Pamięć jest zwalniana w destruktorze korzystając z operatora **delete**.

Do wykonania podstawach operacji („+”, „-”, „[]”) wykorzystamy **przeciążone operatory „+”, „-”, „[]”**.

Do wyświetlenia współrzędnych proszę użyć przeciążonego operatora **„<<”**. Należy również zdefiniować operator **„=”.**

Dla klasy należy zaimplementować **konstruktor właściwy, który pobiera jako parametry rozmiar wektora N i tablicę współrzędnych wektora (*float), oraz konstruktor kopiujący i destruktor**.

Zadania w programie głównym:
1. stworzenie dwóch wektorów (przy użyciu obydwu konstruktorów) i wyświetlenie informacji o nich (współrzędne wektorów mogą być podawane przez użytkownika z klawiatury lub wpisane od razu w programie) **2 p**
2. dodanie wektorów do siebie (przy równoczesnym użyciu operatora “=”) i wyświetlenie wektora wynikowego **1p**
3. odjęcie wektorów do siebie i wyświetlenie wektora wynikowego **1p**
4. dostęp do elementu wektora na podanym przez użytkownika miejscu za pomocą „[]” **1p**
Wszystkie operacje należy sprawdzić wypisując wektor na ekran.

Operatory +, -, *, /
Mogą być przeładowane w wersji jedno- [wewnątrz klasy] lub dwuargumentowej [na zewnątrz klasy]. Tak jak w przypadku każdej innej funkcji nie istnieją ograniczenia co do zwracanych wartości: operator taki może zwracać obiekt tej samej klasy, którą dodaje (np. możemy napisać klasę która dodając do siebie dwa obiekty klasy człowiek zwróci również człowieka), lub dowolny inny, jaki sobie wymyślimy (dodając do siebie dwóch ludzi zwróci nam np. sumę ich wieku).

Operator przypisania = (za wykładem)
Dwuargumentowy operator przypisania = służy do przypisanie jednemu obiektowy klasy treści drugiego obiektu tej samej klasy.
Jeśli operator= nie zostanie zdefiniowany, to zostanie on automatycznie wygenerowany (**podobnie jak konstruktor kopiujący**!), przepisane zostaną pola “składnik po składniku” (podobnie jak w przypadku automatycznie generowanego konstruktora kopiującego). W rezultacie takiego przypisania będą dwa obiekty o bliźniaczej treści.
Kłopoty natomiast mogą pojawić się wtedy, kiedy **składnikami klasy są wskaźniki** lub jeśli klasa używa **operatora new do rezerwacji miejsca w zapasie pamięci**.
Operator przypisania składa się z części:
(1) sprawdzenie, czy nie jest kopiowane siebie samego
(2) części “destruktorowej” (np. zwalnianie pamięci)
(3) części “konstruktorowej”, przypominającej konstruktor kopiujący
Przykład użycia: **(jeśli operator znajduje się wewnątrz klasy):**

```
Nazwa & operator=(Nazwa & na){;
    if (&na == this) return *this; //sprawdzenie, czy nie kopiuje samego siebie
    delete X; //zwalniamy pamięć dla rzeczy, które mogły mieć poprzednio zaalokowaną pamięć
   //alokujemy nową pamięć i przepisujemy wartości tak samo jak w konstruktorze kopiującym
    return *this;
 }
```

Operator [ ] to operator odwołania się do elementu tablicy, jest dwuargumentowy. Chcąc napisać funkcję operatorową [ ], tak, aby operator [ ] mógł stać po obu stronach znaku = musimy zadeklarować, że funkcja ta będzie zwracała referencję do tego, czemu mamy przypisywać!
Przykład implementacji operatora [ ]:

```
int operator[](int i) { return a[i]; }
```