Тип псевдоним

1. **Задаване на тип псевдоним**

Нека T е име на тип. Типът **T&** е тип псевдоним на T.

T се нарича базов тип на типа псевдоним.

2. **Множество от допустими стойности** – всички имена на дефинирани вече променливи.

Пример: Нека програмата съдържа следните дефиниции

int a, b = 5;

Множеството от стойности на типа **int&** съдържа имената a и b. Променлива величина, множеството от допустимите стойности на която съвпада с множеството от стойности на даден тип псевдоним, се нарича променлива от този тип псевдоним.

3. **Дефиниране**

<дефиниция_на_променлива_от_тип_псевдоним> ::=

T& <идентификатор> = <вече дефинирана променлива>; |

T &<var> = <defined_var_of_T>;

където

T е име тип, а

<defined_var_of_T> е име на вече дефинирана променлива от тип T.

Нарича се инициализатор.

Пример: Дефинициите

```
int a = 5;
int & s = a;     //s- псевдоним на a
double r = 1.85;
double &s1 = r, &s2 = r;          //s1 – псевдоним на r, s2- също
int& s3 = a, s4 = a;       //s3- псевдоним на a
```

Дефинициите задължително са с инициализация – променлива от същия тип като на базовия тип на типа псевдоним.

Освен това, след инициализацията, променливата псевдоним не може да се променя като й се присвоява нова променлива или чрез повторна дефиниция. Затова тя е "най-константната" променлива, която може да съществува.

Пример:

```
...
int a = 5;
int &s = a;  // s е псевдоним на a
int b = 10;
int& s = b;                  // error, повторна дефиниция
...
```

4. Операции и вградени функции - всички операции и вградени функции, които могат да се прилагат над първообраза, могат да се прилагат и над псевдонима й и обратно.

Примери:

1.
```
int a = 5;
int &syn = a;
cout << syn << "  " << a << '\n';
int b = 10;
syn = b;
cout << b << "  " << a << "  " << syn << '\n';
```

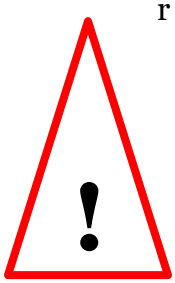
извежда

```
5      5
10     10   10
```

Операторът syn = b; е еквивалентен на a = b;.

2.
```
int i = 1;
int& r = i;    // r и i са свързани с едно и също цяло число
cout << r; // извежда 1
int x = r;     // x има стойност 1
```

r = 2; // еквивалентно е на i = 2;

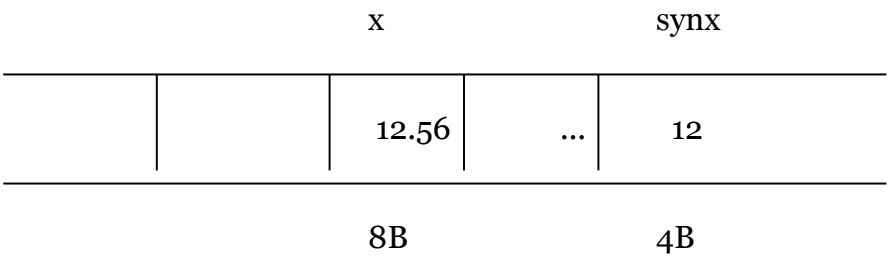


Допълнение: Възможно е типът на инициализатора да е различен от този на псевдонима. В този случай се създава нова, наречена **временна**, променлива от типа на псевдонима, която се инициализира със зададената от инициализатора стойност, преобразувана до типа на псевдонима.

Например, след дефиницията

```
double x = 12.56;
int& synx = x;
```

имаме



Сега x и псевдонимът й synx са различни променливи и промяната на x няма да влияе на synx и обратно.

5. Константни псевдоними

В C++ е възможно да се дефинират псевдоними, които са константи. За целта се използва запазената дума const, която се поставя пред дефиницията на променливата от тип псевдоним. По такъв начин псевдонимът не може да променя стойността си, но ако е псевдоним на променлива, промяната на стойността му може да стане чрез промяна на променливата.

Пример: Фрагментът

```
int i = 125;
const int& syni = i;
cout << i << " " << syni << '\n';
syni = 25;
cout << i << " " << syni << '\n';
```

ще съобщи за грешка (syni е константа и не може да е лява страна на оператор за присвояване), но фрагментът

```
int i = 125;
const int& syni = i;
    cout << i << " " << syni << '\n';
    i = i + 25;
    cout << i << " " << syni << '\n';
```

ще изведе

125 125

150 150

Последното показва, че константен псевдоним на променлива защитава промяната на стойността на променливата чрез псевдонима.

Темата е направена по учебника на Магдалина Тодорова – Програмиране на C++