Псевдоним, упражнение

гл.ас. д-р. Нора Ангелова

Приоритет на операциите

- ++, -- (постфиксни)
- !, +, -, ++, -- (унарни, префиксни)
- *, /, %
- +, - (бинарни)
- >> << (вход/изход)
- <=, >, >=
- ==, !=
- &&
- | |
- =

Какво ще изведе

```
int i = 1;
int a = ++i + 2;

cout << a;
```

Резултат: 4

Какво ще изведе

```
int i = 1;
int a = i++ + 2;

cout << a;
Резултат: 3

cout << i;
Резултат: 2
```

Псевдоним

```
int i = 1;

int * pointer = &i;
int& ps = i;
cout << ps << *pointer;
```

11

```
ps = 5;
cout << ps << *pointer;
```

55

```
int j = 3;
ps = j;
cout << ps << *pointer;
```

33

Псевдоним

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

5 5

10 10 10

Константен псевдоним

```
int i = 25;
const int& syni = i;
cout << i << " " << syni << '\n';
syni = 50; // Грешка!!! syni е константа
cout << i << " " << syni << '\n';
```

25 25

Константен псевдоним

```
int i = 25;
const int& syni = i;
cout << i << " " << syni << '\n';
i = 50;
cout << i << " " << syni << '\n';
```

25 25

50 50

Какво ще изведе

```
int f(int& x)
{
  x++;
  return x;
}

int main()
{
  int a = 5;
  cout << f(a) << endl;
  return 0;
}
```

6

Какво ще изведе

```
int f(int& x)
{
  return ++x;
}

int main()
{
  int a = 5;
  cout << f(a) << endl;
  return 0;
}
```

6

Какво ще изведе

```
int f(int& x)
{
  return x++;
}

int main()
{
  int a = 5;
  cout << f(a) << endl;
  return 0;
}
```

5

Какво ще изведе

```
int f(int& x)
{
  return ++x;
}

int main()
{
  int a = 5;
  cout << f(a+1) << endl;
  return 0;
}
```

Error: cannot convert parameter 1 from 'int' to 'int &'

Какво ще изведе

```
int f(const int& x)
{
  return x;
}

int main()
{
  int a = 5;
  cout << f(a+1) << endl;
  return 0;
}
```

6

Какво ще изведе

```
char str[10];
char str2[10];

cin.get(str, 10, ' ');
cin.get(str2, 10, ' ');

cout << str << endl;
cout << str2 << endl;
```

вход: 1234 567

1234

празно

Какво ще изведе

```
char str[10];
char str2[10];

cin.getline(str, 10, ' ');
cin.get(str2, 10, ' ');

cout << str << endl;
cout << str2 << endl;
```

вход: 1234 567

1234

567

Какво ще изведе

```
char str[10];
char str2[10];

cin.get(str, 10, ' ');

char s1;
cin >> s1;

cin.get(str2, 10, ' ');

cout << str << endl;
cout << str2 << endl;
```

вход: 1234 567

1234
67

Какво ще изведе

```
char str[10];
char str2[10];

cin.get (str, 10, ' ');

char s1;
cin.get(s1);

cin.get(str2, 10, ' ');

cout << str << endl;
cout << str2 << endl;
```

вход: 1234 567

1234

567

Рекурсия

Задача

Дадено е неотрицателно цяло число. Да се дефинира рекурсивна функция, която намира броя на цифрите на числото в бройна система с основа k.

```
int countDig(int n, int k)
{
  if (n < k)
  {
    return 1;
  }

  return countDig(n/k, k) + 1;
}
```

Рекурсия

Задача

Да се дефинира рекурсивна функция, която заменя всяко срещане на цифрата 5 в дадено неотрицателно цяло число с 8.

```
int replace(int n)
{
  if (n == 0)
  {
    return 0;
  }

  int lastDigit = n % 10;
  if (lastDigit == 5)
  {
    return replace(n/10)*10 + 8;
  }

  return replace(n/10)*10 + lastDigit;
}
```

Рекурсия

Задача

Лабиринт е представен с булева квадратна матрица 8x8. Клетка се приема за проходима, ако елементът в съответната позиция е истина и за непроходима в противен случай.

Да се напише програма, която проверява дали съществува път от съседни в хоризонтално и вертикално направление проходими клетки на лабиринта, който започва в горния му ляв ъгъл и завършва в долния му десен ъгъл.

Рекурсия

```
bool labyrint[8][8] = {
  {1, 0, 0, 0, 1, 1, 1, 1},
  {1, 0, 0, 0, 1, 1, 1, 1},
  ...
  {1, 0, 0, 0, 1, 1, 1, 1},
};

bool way(int x, int y)
{
  // напуснали сме границите на лабиринта
  if (x < 0 || y < 0 || x > 7 || y > 7)
    return false;

  // целта е достигната
  if (x == 7 && y == 7)
    return true;

  // клетката е непроходима
  if (!labyrint[x][y])
    return false;

  // обявяваме клетката за обходена, за да се предотврати зацикляне
  labyrint[x][y] = 0;

  //търсене на път от някой от четирите съседа
  return way(x+1, y) ||
    way(x, y+1) ||
    way(x-1, y) ||
    way(x, y-1);
}
```

```
cout << “Край”;
```