**I.1) Instrukcja warunkowa if**

*if (warunek):*

*(polecenie)*

Np.:

if imie == "Jan" and wiek == 23:

print "Nazywasz sie Jan i masz 23 lata."

if imie == "Jan" or imie == "Robert":

print "Nazywasz sie Jan lub Robert"

**I.2) Implementacja operatora warunkowego ?**zamiast jak w c/c++:  
 *(warunek ? prawda : fałsz)*

W pythonie mamy:

*(prawda if warunek else fałsz)*  
Np.:

wynik = x if a > b else y

**I.3) Pętla while oraz pętla for … in**

Pętla while sprawdza czy warunek jest prawdziwy, i jeżeli tak – wykonuje polecenie, i zapętla się aż do momentu gdy warunek przestanie być prawdziwy.  
 *while (warunek):  
 (polecenie)*  
Np.:  
 while x<10:  
 print(x)  
 x += 1

Pętla for … in … wykona się n razy, n zdefiniowane jest przez ilość elementów drugiego argumentu (lub bezpośrednio: *funkcja range()*)  
 *for (zmienna) in (zakres):  
 (polecenie)*  
Np.:  
 for znak in slowo:  
 print(znak)

**I.4) Jaka jest różnica między break a continue?**  
break – „łamie” pętle, coś jak goto do pierwszego miejsca poza pętle  
continue – „pomija” obecną iteracje pętli i wraca do początku  
  
**I.5) Jak się deklaruje funkcje w Pythonie?** *def nazwaFunkcji(argumenty):  
 (polecenia)  
 (return)*  
Np.:  
 def mnożenie(x,y=1): #jeżeli nie podamy y to y=1 domyślnie  
 z = x\*y  
 return z;  
  
**I.6) Funkcja lambda**  
Funkcja (wyrażenie) lambda to taka jednolinijkowa mini funkcja, która może przyjmować wiele argumentów  
 g = lambda x: print(x\*2)  
 g(10) # zwróci 20

**II.1) Domyślna wartość argumentu**  
Spodziewałem się wyniku 5 (odwołania się funkcji do obecnej wartości a w momencie wywołania funkcji), jednak ze względu na to że w Pythonie zmienna to bezpośrednie wskazanie na dane, w momencie definicji funkcji argument a przybiera wartość 2 i „zakorzenia się” w definicje funkcji funkcja zwraca 2.

**II.3) Zmienne lokalne**Bezpośrednio wynikiem wywołania f3(x) będzie błąd – x nie został nigdzie zadeklarowany, jeżeli jednak wykonamy f3(w) to otrzymamy:  
 zmienna 'w' ma wartość 10  
 lokalnie 'w' zmieniono na 12   
 ale globalnie, 'w' ciągle wynosi 10  
Możemy zobaczyć, że nadpisanie w obowiązuje jedynie w funkcji.

**II.4) Zmienne globalne**Wynik działania:  
 zmienna 'w' ma wartość 10  
 lokalnie 'w' zmieniono na 12  
 ale globalnie, 'w' ciągle wynosi 12  
Poprzez deklaracje **global w** „włączamy” tryb globalnego edytowania zmiennej (zapisywanie zmian poza funkcją)