1. Теория.

1.1. Преобразование типов данных. Ошибки присвоения и ошибки округления.

int(a [,kind]) – преобразует а (целое, вещественное, комплексное) в целый тип с разновидностью kind путем отсечения значений а в сторону 0. а может быть массивом. В случае комплексного типа а, преобразуется только вещественная часть. На самом деле, тоже самое значение будет при простом присваивании типа i=a, если i – integer, а а – real или complex.

real(a [,kind]) – тоже самое, для перевода в действительный тип данных. Аналогичный результат для a=c, если c – integer или complex, а а – real.

dble(a) – исключительно в real(8).

cmplx(x [,y] [,kind]) –аналогично в комплексный тип. Необязательный параметр y может добавить мнимые значения. х и у могут быть согласованными массивами. В упрощенным виде также работает присваиванием.

dcmplx(x [,y]) – исключительно в complex(8).

Такая возможность заменять операторы присваиванием может приводить к ошибкам в присваивании.

Существует проблема с определением для real(8)

ca=-2.300000000000000

i=5

ca2=i\*ca

print \*, ca2 ! -11.4999997615814

Все дальнейшие вычисления идут, как кажется, с ошибкой округления. Для real(4) проблема исчезает, но точность ограничена 6 знаком после запятой. Последняя цифра определяется округлением. Так:

pi=3.14159265358979

для real(4) даст результат 3.141593. а real(8) - 3.14159274101257.

Если речь идет о числах близких к нулю, то проблема решается с помощью десятичного представления.

a=1E-9

Впрочем, данная проблема является скорее проблемой вывода данных:

ca2=-2.300000000000000

ca3=0

do i=1,10000

ca3=ca3-2.300000000000000

end do

ca3=ca3/10000

print \*, ca2

print \*, ca3

В данном примере са2 и са3 имеют одинаковые значения, что говорит о том что ошибка округления не накапливается. В целом, используя десятичное представление можно двумя числами вывести результат с любой точностью. Для real(8) возможен правильный счет с точностью до 45знака.

Возможны довольно хитрые ошибки. Например, если а – целый массив, а i,j – integer.

do j=1,2

do i=1,5

a(i,j)=i/(10\*j)

end do

end do

Даст одни нули. Так как справа все элементы целые, присвоение идет после округления. Проблема решается если переписать присвоение как:

a(i,j)=i/(10.0\*j)

Вообще стоит взять за правило присваивать величины разных типов только после преобразования типа.

1.2. Перезагрузка и родовые имена.

Можно объединить несколько процедур одним родовым именем, это называется перегрузкой процедур.

Сделаем процедуру нахождения максимума из 2 чисел для двух типов данных сразу:

interface mymax

function inmax(i1,i2)

integer(4) inmax, i1, i2

end function inmax

function remax(a1,a2)

real(4) remax, a1, a2

end function remax

end interface

function inmax(i1,i2)

integer(4) inmax, i1, i2

if (i1 >=i2) then

inmax=i1

else

inmax=i2

end if

end function inmax

function remax(a1,a2)

real(4) remax, a1, a2

if (a1 >=a2) then

remax=a1

else

remax=a2

end if

end function remax

Такая процедура использует функцию inmax для целых и remax для действительных.

Можно создать аналогичную процедуру как модуль.

Аналогично можно перегрузить операцию, в том числе уже существующие.

2. Задание.

1.1. Преобразование типов данных. Ошибки присвоения и ошибки округления.

Придумать задачу, требующую преобразования данных, избегая ошибок присваивания и округления.

1.2. Перезагрузка и родовые имена.