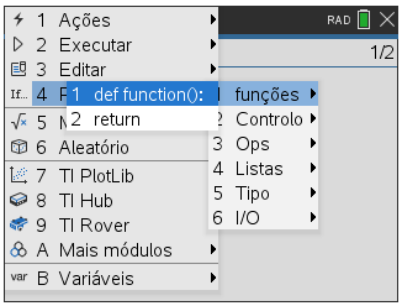
```
def modulo(x):
    if x<0:
        return -x
    else:
        return x
x=float(input('x = '))
print('A imagem de ',x,' é: ',modulo(x))
```

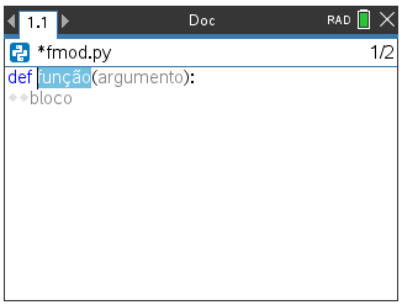
1. Como determinar a imagem de um objeto com a função módulo?

I. Para dar resposta, em Python, a esta questão, pode construir-se um programa com uma estrutura condicional, com duas condições, analogamente ao que foi feito no programa relativo à fórmula resolvente da equação do 2º grau.
No entanto, pode considerar-se esse programa numa estrutura de função em *Python*. Esta "função" é semelhante ao que noutras linguagens de programação se chama de subrotina, ou seja, uma secção de código que é ativada a partir das linhas de código do programa subsequentes. Esta estrutura é bastante útil, por exemplo, quando no programa se precisa de fazer algumas vezes as ações que nela estão previstas. A ideia desta estrutura é oportuna num contexto de funções matemáticas e, por isso, surge aqui como exemplo de um passo mais à frente em relação à simples utilização de uma estrutura condicional.

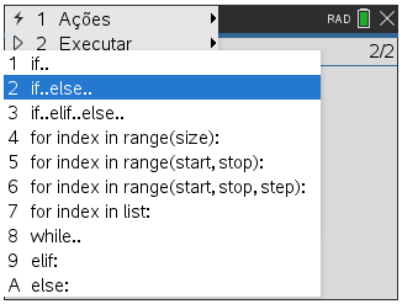


Para obter esta estrutura, caso conheça a sintaxe, poderá escrever com o teclado, mas pode também obter através do menu:

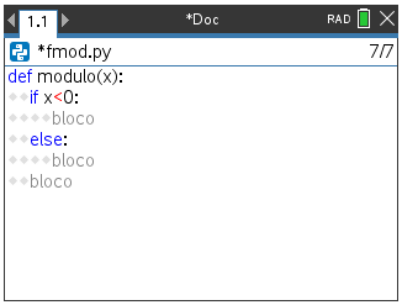
menu 4 **Planos integrados** → 1 **funções** → 1 **def function():**



Obtida a estrutura, é agora necessário preencher os campos em falta.

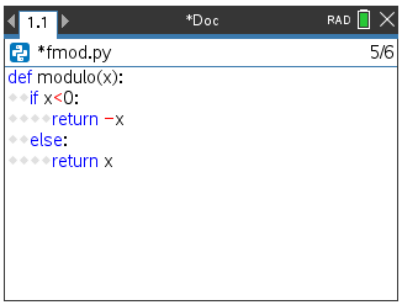


II. Nos espaços em falta deve começar-se por nomear a função, nome pelo qual será ativada no programa. Aqui escolheu-se "*modulo*".
No bloco vai colocar-se o que se pretende que esta função faça, ou seja, que devolva o simétrico do número que entra como argumento ou, caso contrário, que devolva o mesmo valor. Por isso, é importante colocar um nome entre parênteses na 1ª linha, para a completar, pois é essa a designação com que vai ficar o dado introduzido e que deverá depois ser considerado nas linhas de código desta estrutura. Neste caso, escolheu-se "x". É, por isso, adequado, considerar uma estrutura de condição *if..else*.



III. Para que, depois da ação realizada por esta secção de código sobre o dado da entrada, o retorno ao programa principal se faça com o resultado, deve escrever-se *return()* com o que se pretende que seja o resultado, a saída desta secção de código.

Este código pode ser facilmente escrito com o teclado, ou obtida no menu, na mesma lista pendente onde se obteve a função *def function():*. Da estrutura base, deve apagar-se o que não for necessário para esta secção funcionar conforme o pretendido.

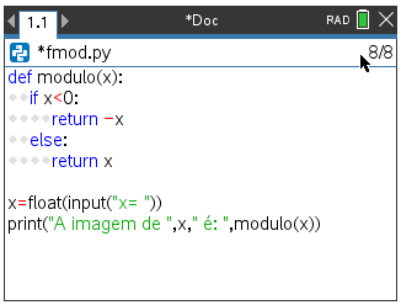


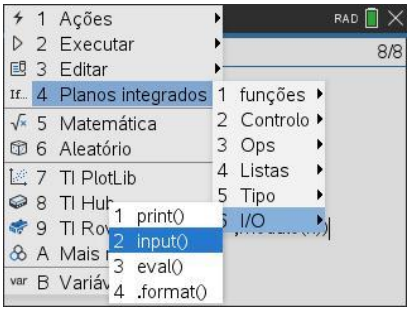
IV. Falta agora escrever o programa principal, o qual vai ter apenas duas linhas, uma para que o utilizador introduza o número que pretende ver devolvido como módulo, e na segunda linha a "chamada" da secção para que devolva o valor do módulo e o escreva.

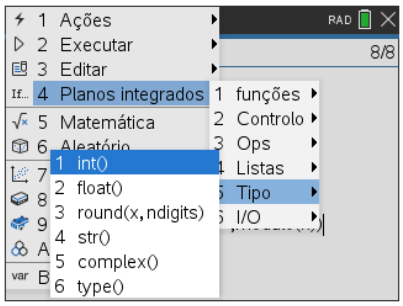
Nestas linhas de código são utilizadas as funções *input()*, a qual aguarda a introdução do dado pelo utilizador quando executar o programa, *float()*, que indica que esse dado é um número, que pode ser inteiro ou não, e *print()*, que se destina a mostrar ao utilizador a saída com o resultado da execução. Estas linhas de código podem escrever-se facilmente com o teclado, mas podem também ser obtidas no menu.

input() e *print()*: menu 4 **Planos integrados** → 6 **I/O** → 2 **input()** / 1 **print()**

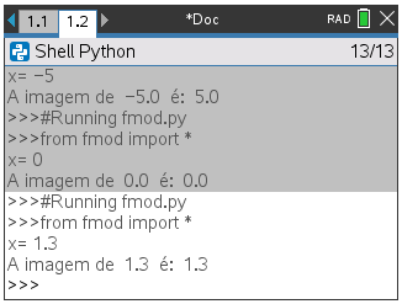
float(): menu 4 **Planos integrados** → 5 **Tipo** → 2 **float()**







V. Escrito o programa, falta executá-lo.



Algumas ideias sobre programação, relacionadas com o contexto