RESUMO

Discutem-se as objectivas dum curso introdutório a programação. para alunos não informáticos. Defende-se que o ensino da programação deve dar especial ênfase a analise do problema e deve incidir sobre mais do que uma linguagem de programação, procurando criar esquemas mentais flexíveis e duradouros que sejam um suporte eficiente para o futuro. Salienta-se a importância, presente e futura, das calculadoras programáveis de bolso e, após discussão das várias alternativas, defende-se o ensino em simultâneo do BASIC E AOS. São descritas as experiências dos autores neste domínio e analisados os resultados obtidas.

Defende-se ainda a necessária separação entre o ensino de programação e da análise numérica, disciplinas de objectivos e métodos distintos. Salienta-se também a oportunidade duma formação básica em informática para futuros profissionais de engenharia.

(')Departamento de Engenharia Química da FEUP e RAR (Porto)
(")Departamento de Engenharia de Minas de FEUP

1.OBJECTIVO DO ENSINO DA PROGRAMAÇÃO

Para um aluno dum curso não informático (engenharia por exemplo) uma cadeira de introdução à programação deve ter um objectivo Simples ensina-lo a usar um computador como "ferramenta" de trabalho, em particular torna-lo apto a tirar partido da Potencialidade que um computador digital oferece para resolver os problemas que a vida profissional certamente lhe irá colocar. Por isso, uma cadeira deste tipo deve evitar longas divagações sobre a estrutura interna dum computador digital, especialmente a nível de hardware, e fazer antes incidir a sua atenção sobre os mecanismos de resolução de problemas com recurso a um computador.

A resolução de um problema passa por uma fase inicial de analise a concepção dum algoritmo de resolução de um problema e sua descrição O recurso a um computador para a resolução do problema passa ainda pela descrição codificada da algoritmo numa linguagem de programação. A maior dificuldade da aprendizagem não reside habitualmente na segunda fase, pelo menos ao nível de linguagem de nível superior mas sim na fase de concepção e descrição do algoritmo, a qual e em larga medida Independente da linguagem especifica da codificação. Logo, é sobre a analise que deve incidir uma atenção especial e será claramente errado reduzir o ensino da programação à aprendizagem simples da gramática duma linguagem especifica. Esta só faz sentido quando integrada no processo de construção, teste e operação de algoritmos. Aliás numa fase inicial de aprendizagem, a codificação do algoritmo é mais ou menos imediata ao nível duma linguagem de nível superior.

Neste trabalho discutem-se alguns princípios que devem presidir a construção de uma cadeira deste tipo. Mais do que o ensino exaustivo duma linguagem, defende-se o ensino dos princípios gerais de mais do que uma linguagem, aprendidas e usadas em simultâneo. Entre as linguagens possíveis, defende-se a utilidade pedagógica e profissional do ensino simultâneo da programação BASIC e em ADS (calculadoras programáveis de bolso). Finalmente rebate-se um principio muito generalizado no ensino universitário português: o ensino, numa mesma cadeira, da programação e da análise numérica.

2. MECANISMOS DA APRENDIZAGEM DA PROGRAMAÇÃO

A elaboração de esquemas mentais (estruturas em que os conceitos são Integrados e interrelacionados) é a base de toda a aprendizagem (1) e permite a integração futura de novos conceitos no conhecimento do indivíduo. A compreensão do um conceito é precisamente a assimilação desse conceito num esquema apropriado. Note-se que a reordenação interna de um esquema pode permitir uma mais fácil apreensão de novos conceitos e facilitar a aprendizagem.

A compreensão dum novo conceito implica portanto a existência dum esquema mental anterior ao qual seja possível fazer a ligação. Este tipo de aprendizagem é diferente da aprendizagem por memorização e, embora mais lento e difícil (em especial se os esquemas anteriores forem inapropriados), garante uma maior persistência dos conhecimentos e prepara o aluno para o futuro. garantindo-lhe uma base sólida donde partir para novos avanços na aprendizagem e na adaptação a novos problemas.

Um esquema mental baseado numa importância excessiva de conceitos acessórios não pode ser flexível e facilmente adoptável. Por isso, o ensino de uma linguagem de programação não deve basear-se no estudo sistemático de todos os seus pormenores (regras e excepções), mas deve centrar-se sobre os conceitos básicos da organização da linguagem e criar condições para que os conceitos e regras acessórios sejam facilmente integrados pelo trabalho independente do aluno, a medida que as circunstancias o forem exigindo.

Mas este processo dinâmico de aprendizagem esquematiza implica resistências psicológicas que não podem ser ignoradas. Um esquema teve um certo custo (esforço) de construção e a resistência a modificações pode ser grande, e será tanto maior quanto mais cristalizado estiver o esquema e menos flexível for a incorporação de novos conceitos. Se esta implicar uma grande reordenação estrutural do esquema anterior, o esforço psicológico podara ser violento e acabar mesmo por rejeitar o novo conceito. Este mecanismo explica a bem conhecida resistência por parte daqueles que se julgam já suficientemente conhecedores. As suas consequências são particularmente importantes na aprendizagem da programação, onde é também conhecido a frequente resistência oferecida ao estudo de uma segunda linguagem. Para que tal não aconteça o processo deve conduzir a esquemas mentais pouco ligados a uma dada linguagem especifica, mas sim baseados em princípios gerais.

Não existem grandes diferenças de fundo entre a maior parte das linguagens (algorítmicas) de programação de nível superior: os princípios da codificação ao semelhantes . pelo menos ao nível das operações elementares (atribuições, decisões, ciclos, etc.). O ensino simultâneo de mais do que uma linguagem é por isso perfeitamente viável e permite a elaboração de esquemas mentais mais gerais, logo mais flexíveis e adaptáveis, menos facilmente cristalizáveis e por isso mais produtivos. Este facto foi exemplarmente reconhecido por Gerald Winberg. A citação (de 1971) será porventura longa, mas talvez valha bem a pena: “This situation could be improved if w'e could enunciate and teach certain principies that are not tied to particular languages, so that even the beginner would have some less relative measure to hold up against the language he is learning. But teaching practice today in our universities and programming schools seems to be pointing in exactly time opposite direction. Instead of trying to teach principles, or at least trying lo teach two contrasting programming 1anguages simultaneously, or at the very least trying to teach one language which rather broadly represents the major possibilities, the schools seem devoted to teaching how to program in a single simple and artificial language. The objective seems to be to get the student writing some kind of program as soon as possible - a not unworthy aim - but at the expense of limiting the future growth of the programmer"(2).

3.QUE LINGUAGENS ENSINAR?

Os princípios que devem presidir a escolha das linguagens a ensinar foram enunciados no capítulo anterior. Far-se-á agora uma breve analise das linguagens de uso mais generalizado, sob o ponto de vista descrito.

O FORTRAN não será uma linguagem aconselhável para um curso de introdução à programação, apesar da divulgação que entre nós tem conhecido, com esse objectivo. E uma linguagem onde as excepções constituem a regra e com instruções input/output muito complexas, que quase constituem uma nova linguagem que não é central para os objectivos de aprendizagem e que distrai do que realmente é importante.

A afinação e correcção de programas em FORTRAN é também morosa e difícil. com consequências nefastas para a eficiência da aprendizagem. Refira-se a ausência do controlo sobre o valor actual das variáveis, a falta de controlo sobre os índices das variáveis indexadas e sobre o tipo de variáveis na transferência do parâmetros em subrotinas. Tudo isto torna difícil a detecção de erros em "run time".

Muitas vezes o FORTRAN funciona, pelo menos entre nós, em sistemas que operam em batch, usando cartões ou fitas perfuradas, com todas as dificuldades de acesso a maquina e morosidade de correcções que implicam. Um elevado tournaround-time tem um efeito profundamente desmoralizador sobre o aluno. Mas se o FORTRAN estiver a operar num sistema em ambiente de tempo distribuído, então o arquivo de programas e dados far-se-á provavelmente em disco com acesso por terminal . A complexidade de utilização do terminal (operações de editing , compilação, etc.) é uma vez mais prejudicial a fase inicial de aprendizagem. e constitui por si próprio uma linguagem de operação do sistema que o aluno tem que aprender e de cujo sucesso depende a aprendizagem da linguagem de programação propriamente dita. E sem acesso fácil e abundante a maquina qualquer ensino de programação fica seriamente prejudicado. Alguns destes inconvenientes de operação são comuns a outras linguagens de programação (ALGOL, por exemplo). Mais recentemente. começaram e surgir no mercado micro-computadores FORTRAN em ROM, que obviem algumas destas dificuldades. embora se mantenham as anteriores.

O ALGOL tem o sério inconveniente de ser pouco disponível, embora seja uma linguagem mais lógica e natural que o FORTRAN. Tem a eventual vantagem de ser uma linguagem estruturada, em que a descrição da sintaxe é mais simples de formalizar e de utilizar de uma maneira formal. O PASCAL é uma linguagem estruturada mais simples e mais potente que o ALGOL. Embora o seu futuro seja muito promissor, é ainda muito recente e pouco disponível.

O APL continua a ter sérios problemas de disponibilidade. Como linguagem introdutória tem ainda o sério problema da simbologia, que a torna muito pouco transparente. Exige ainda a introdução de novas operações. É também difícil fazer a passagem do APL para outras linguagem.. a que. de acordo com o anteriormente exposto é um sério contra-objectivo para um curso de introdução á programação.

O BASIC tem vindo a ser cada vez mais divulgado para o ensino da programação, aliás de acordo com os objectivos que presidiram á sua elaboração. É uma linguagem simples e largamente disponível, que resume o essencial para a descrição de algoritmos e permite uma fácil extensão natural a grande parte das outras linguagens mais elaboradas (FORTRAN por exemplo). O recurso á codificação em BASIC permite concentrar o esforço na construção do algoritmo: o essencial continuara a ser a análise e não a codificação.

O facto do BASIC ser normalmente usado com interpretador tem vantagens ao nível de aprendizagem, permitindo um fácil "debugging". Para isso contribui ainda as possibilidades de HALT/CONTINUE com eventual alteração de variáveis e interrogação sobre os seus valores.

Extensões do BASIC tem também conhecido uma larga divulgação para além das aplicações cientificas, em especial na gestão média (será provavelmente a linguagem hoje em dia mais divulgadas, logo a seguir ao RPG).

1.A PROGRAMAÇÃO DE CALCULADORAS DE BOLSO

As calculadoras programáveis de bolso tem vindo a conhecer uma popularidade crescente. ao mesmo tempo que a sua potência de calculo vai aumentando. A sua portabilidade fácil e baixo custo, as características de acesso imediato. interactivo e pessoal dão-lhe uma importância na vida profissional que não pode ser ignorada. As limitações de memória os elevados tempos de processamento obrigam a um esforço de optimização dos programas, mas, por outro lado, a facilidade de acesso prolongado a uma máquina deste tipo faz perder muito do carácter negativo que os elevados tempos de calculo implicam.

Una análise sistemática das principais características das calculadoras programáveis de bolso foi já tentada em trabalhos anteriores (3,4). Recorda-se um certo grau de especialização, uma memória de programa/dados de reduzida capacidade. a possível existência de uma biblioteca de programas em circuitos integrados extra-memória, que em futuro próximo poderão ser escri tos pelo próprio utilizador, um tempo de execução típico de 10 multiplicações por segundo. A linguagem do programação é

simplificada, do tipo assembler. mas a unidade aritmética é muito sofisticada. No entanto, o assembler é dotado de instruções sofisticadas (variadas funções por hardware, processamento em vírgula flutuante por hardware, instruções de transferencia e de operação directa sobre a memória com endereçamento indirecto, etc.).

As potencialidades das calculadoras programáveis de bolso para cálculos sofisticados de engenharia foram já demonstradas em vários trabalhos anteriores (3.5.6). O ensino não pode ficar alheio a esta revolução no domínio dos meios baratos de calculo e deles tem que saber tirar partido (7). A linguagem de programação de calculadoras de bolso (ADS e HPN são as mais divulgadas) não só tem interesse imediato e futuro para os alunos, como serve de forma exemplar para complementar a aprendizagem da programação em BASIC. O caracter mais elementar das instruções e a menor transparência do programa não são propriamente inconvenientes. O conhecimento da programação dum computador em assembler é de pouca utilidade futura, dada a sua dependência de um equipamento especifico. No entanto, ajuda a clarificar muitos conceitos de programação. Um caso exemplar será o uso de variáveis indexadas. Embora o conceito de "array" não seja difícil de apreender, o seu uso correcto é uma dificuldade crónica da aprendizagem da programação. o acesso indirecto a posições de memória uma calculadora programável de bolso ajuda imenso a clarificar o conceito.

- A aprendizagem simultânea de BASIC e ADS (ou HPN) permite criar um feedback mutuo de que resulta uma notável economia do esforço global de aprendizagem a uma menor tendência ; cristalização dos esquemas formados. A reacção do aluno a este tipo de ensino é também francamente favorável.

A escolha entre ADS e HPN é difícil. A nossa experiência tem sido feita com ADS por razões de facilidade de acesso. O equipamento Texas Instruments é, de longe, o mais divulgado e barato em Portugal e a quase totalidade dos alunos dispõe de máquinas desta marca. As máquinas Hewlett-Packard funcionam tradicionalmente em HPN, mas são pouco divulgadas entre nós.

- O ADS está mais na linha de escrever expressões do que calcular o seu valor numérico, e talvez por isso os alunos sintam uma maior facilidade na sua introdução. No entanto o HPN será mais natural, na linha da primitiva maquina de calcular para uso contabilistico (onde predominam hábitos de calculo, mais do que de escrita de expressões). Refira-se, de passagem, que o ADS tem incoerências (por exemplo, a codificação de ln (A+B) será (A+B) ln).

5, EXPERIÊNCIA DO ENSINO SIMULTÂNEO DE BASIC E ADS

Há já vários anos que a FEUP tem vindo a basear o ensino de programação no BASIC, quebrando uma tradição de FORTRAN prevalecente até ai na Universidade do Porto (consequência do tipo de equipamento do único centro ate ai disponível: o LACA da FCUP). Apesar das limitações de equipamento com que a FEUP se vem cronicamente debatendo, os resultados tem sido encorajadores.

A introdução informal das calculadoras de bolso e sua programação começou a ser tentada pelos autores a partir de 1977, em cursos para alunos dos departamentos de Minas e de Química da FEUP. A boa receptividade encontrada levou a que mais recentemente tivesse sido introduzida com carácter de obrigatoriedade, em simultâneo com o BASIC. Para tal contribuiu ainda a experiência positiva dum curso intensivo de programação (9) para quadros técnicos da industria, em que, com abundantes facilidades de acesso a equipamento, se tentou o ensino simultâneo de BASIC, ADS e HPN (embora com maior ênfase no ADS).

A apresentação de conceitos e exemplos de ambas as linguagens é feita em simultâneo. Os problemas práticos apelam à resolução numa ou noutra linguagem, evitando privilegiar qualquer delas. Aos alunos são dadas possibilidades de acesso a calculadoras Texas T159 com impressora PC100B, assim como a vários micro-computadores com interpretadores BASIC.

Um inquérito aos alunos da cadeira de Computação e Métodos Numéricos, do 2º ano de licenciatura em engenharia química da FEUP, permitiu quantificar alguns aspectos da sua receptividade a este tipo de ensino (10).

Cerca de 55% dos 60 alunos declararam possuir uma calculadora programável de bolso. 35% dos alunos eram repetentes. O questionário envolvia a avaliação quantitativa de interesses através da marcação duma posição apropriada numa borra entre o 0 (nenhum) e 1CD (muitíssimo). O inquérito teve lugar no final da 1ª fase da cadeira, após o ensino da programação e antes de se iniciar a parte da analise numérica.

Ointeresse que os alunos encontraram na aprendizagem da programação de calculadoras de bolso (ADS) é muito elevado. O valor médio foi 86.5. mas o histograma (figura 1) apresenta uma distribuição assimétrica no sentido dos valores mais altos. A influencia de "possuir calculadora programável" e "ser repetente" não é significativa. O interesse ao longo do curso e na vida profissional futura tem comportamento semelhante. embora o histograma não seja tão alongado para valores elevados.

O grau médio de dificuldade atribuído ao BASIC é inferior ao atribuído ao ADS (figura 2). Este resultado seria de esperar. atendendo a que o ADS é uma linguagem de nível menos elaborado e por isso menos transparente. Na tabela 1 indicam-se ainda resultados sobre o grau de dificuldade atribuído a. programação em geral (incluindo portanto a análise e codificação de problemas e não apenas o aspecto de linguagem). A influencia de "possuir calculadora programável própria" e "ser repetente" sobre o grau de dificuldade atribuído ao BASIC e ADS continua a não ser significativa.

Mais de 78 dos alunos declararam preferir o ensino em simultâneo das duas linguagens de programação. em alternativa ao ensino separado. Infelizmente. Não é possível dispor de resultados "a branco". mas este resultado parece significativo, em especial se se atender a que o grau de dificuldade atribuída a ADS foi significativamente superior ao atribuído ao BASIC.

A influencia de "possuir calculadora programável própria" não é significativa. No entanto. o grau de dificuldade atribuído a ambas as linguagens. e á aprendizagem da programação em geral. é superior no grupo minoritário dos que preferem um ensino separado das duas linguagens. O grau de dificuldade geral da programação é marginalmente inferior nos alunos repetentes. que tinham passado por uma experiência anterior de programação.

Curiosamente. o coeficiente de correlação entre os graus de dificuldades do BASIC e ADS é quase nulo (r=-0,07). Não há portanto um padrão definido de respostas em torno dos valores médios do grau de dificuldade atribuído a cada uma das linguagens.

8.ENSINO DA PROGRAMAÇÃO E ENSINO DA ANALISE NUMÉRICA

A principal consequência da perniciosa tradição do ensino simultâneo da programação e da análise numérica que os alunos saiem da escola com uma larga confusão entre o que é cada um desses assuntos, sem ter sedimentado seriamente qualquer deles e. ainda por cima. a pensar que ficaram a saber alguma coisa de informática. A origem desta situação reside numa confusão muito generalizada entre os objectivos e métodos da programação, da analise numérica e da informática.

A análise numérica faz apelo à programação, mas o inverso não é verdade. E as únicas relações entre análise numérica e informática residem no facto de tantas fazerem uso do computador, embora com objectivos diferentes. Por outro lado. programação e analise numérica tem metodologias diferentes. que fazem apelo a aptidões diferentes dos alunos. Será talvez oportuno recordar as palavras de Richard Hamning "Teaching programmning along with numerical analysis, as is often tried today, violates a well known pedagogical principle that you should not try to teach more than one new idea at a time. .. As a result, the student emerges with a wrong conception of what numerical analysis is and how to use it pratically".(11).

A melhor solução parece estar num semestre de introdução ; programação logo no inicio do curso, onde se ensinaria o BASIC e o ADS. depois fazer apelo a esses conhecimentos em cadeiras de aplicação (o que pressupõe uma formação dos docentes dessas cadeiras, que infelizmente nem sempre existe) e uma cadeira (semestral) mais a meio do curso sobre análise numérica. Uma solução deste tipo tem vindo a ser

ensaiada desde 1974 no Departamento de Minas da FEUP. Aplicações da programação aprendida durante o 1º ano são incentivadas em cadeiras dos anos seguintes (em espacial de mecânica. onde algumas técnicas numéricas são informalmente introduzidas). e o tratamento matemático de analise numérica é feito no 1º semestre do 4º ano (de futuro, provavelmente no 3º ano). ao que se segue ainda um semestre dedicado á aplicação sistemática de técnicas numéricas à simulação de processos, onde naturalmente se faz largo apelo á programação (em BASIC: e ADS, conforme o tipo de problemas). Lamentavelmente. as propostas no sentido de separar a programação e análise numérica não tem merecido o apoio do departamento de química. Os principais prejudicados são. no fundo os alunos.

Referencias

(1)Skemp. R., "The psychology of learning mathematics". Penguin Books, 1971

(2)Winberg, G., “The psychology of computer programming". Van Nostrand Reishould, Co., 1971.

(3)Regueiras. P.. "Computadores: estrutura. funcionamento, tipos e utilizações". módulo para Beira et al. "Métodos Numéricos em engenharia química,. Porto. 1973.

(4)Madureira. C.. “Potencialidades das calculadoras programáveis de bolso”. modulo para Beira et al. "Métodos Numéricos em engenharia química", Porto. 1978.

(5)Beira. E. e M. R. Costa, "Chemical process optimization with a programable pocket calculator", Congresso Chisa 78, Praga. 1978.

(6)Beira, E.. "Calculadoras de bolso: potencialidades para o projecto de equipamento de transferencia de massa”. módulo para Jerónimo, M.., "Curso de Aperfeiçoamento sobre Transferencia de massa” Porto, 1978.

(7)Beira. E.. "Para uma pedagogia das calculadoras de bolso", 1º Encontro Internacional sobre educação em Química, Lisboa. Outubro de 1975.

(8)Madureira. C., P. Regueiras e E. Beira. "A informática e a FEUP". comunicação ao colóquio "O ensino da informática e a informática no ensino”. API - Norte, Janeiro de 1979.

(9)Beira E., C. Madureira, P. Regueiras e J. Reis, “introdução á programação: calculadoras de bolso e BASIC", Porto, Fevereiro do 1979.

(10)Beira, E.. “Calculadora de bolso e ensino de programação: aspectos de uma experiência". 3º Encontro Nacional de Química, Coimbra, Abril de 1980.

(11)Hammlng, A., "Introdution to applied numerical analysis ”, Mc.Graw Hill, 1971.