Análise do problema

Entendendo um problema. Frequentemente achamos problemas e às vezes temos a necessidade de resolve-lo, como por exemplo, "acabou a água na geladeira, como resolver?". Se formos pensar, alguns problemas possuem solução como por exemplo no caso da água da geladeira é só colocar água dentro da geladeira, já outros não existem solução.

Um problema basicamente persiste de uma pergunta e é de interesse de alguém saber a resposta. Assim, a frase "Acabou a água na geladeira" não é uma pergunta e não pode ser tratado como um problema pois como indica, é apenas um fato e ninguém pode ter interesse em arrumar isso, isto é, mudar o estado de "geladeira sem água" para "geladeira com água". De modo análogo, "Geladeira sem água, como consertar?" é uma pergunta e pode ser tratado com um problema.

Sempre quando deparamos um problema, intuitivamente já elaboramos o plano para resolver de maneira rápida que não nos damos conta desse processo, assim como após aprender a andar de bicicleta não se pensa mais na complexidade de coordenar os movimentos juntos de virar e pedalar.

Contudo, devemos nos atentar se o problema foi entendido corretamente? É inútil tentar resolver um problema que não foi entendido pois neste caso poderá(na maioria dos casos) levar a uma solução incorreta do problema.

Por exemplo, digamos que queremos saber a raiz quadrada de 1764. Então a pergunta seria "Qual é a raiz quadrada de 1764?". Só conseguimos responder essa pergunta se soubermos o que é uma raiz quadrada, não necessariamente como obter o seu valor. Uma vez que sabemos do que se trata, podemos pensar em elaborar um plano, mas de início uma resposta é simples: "A raiz quadrada de 1764 é um número r(positivo) tal que $r^2 = 1764$". Essa seria a própria definição, demos uma resposta. Agora, podemos pensar se essa resposta é satisfatória ou não, aí já entra em outro aspecto, mas é de interesse principal que se entenda o problema.

Elaborar um plano. Uma vez descrito e entendido o problema, queremos achar uma resposta. Mas como achar a resposta? Difícil resposta! Te pergunto, como explicar a alguma pessoa que nunca colocou água na geladeira para arrumar a "Geladeira sem água" para "Geladeira com água"? Da mesma maneira, como ensinar uma pessoa para criar os passos?

(1) Encha um reservatório de água
(2) Abra a porta da geladeira
(3) Coloque a água na geladeira
(4) Feche a porta da geladeira

Essa é uma tarefa difícil e frequentemente o que se faz é retomar a problemas menores que sabemos resolver: conhecemos os comandos de "Encher reservatório de água", "Abrir e fechar a porta" e "Colocar a água na geladeira". E então auxiliar conectando vários passos já conhecidos para montar algo novo de interesse, no caso, deixar no caso "Geladeira com água". Esse processo normalmente utiliza problemas parecidos que já resolvemos, como no caso da raiz de 1764, já achamos qual é a raiz de 4 ou 9 e então podemos usar o mesmo princípio para achar a raiz de 1764.

Assim, o processo de elaboração de um plano é a decomposição em problemas menores e conexão entre estes problemas menores. Vale lembrar que a maneira de conexão é uma das mais(para não dizer a mais) importantes. Uma vez feita a conexão dos problemas menores, temos uma sequência de comandos para fazer(assim como no exemplo acima) e é chamado então de algoritmo.

Executar o plano. Na execução de um plano, pode-se encontrar alguns erros, seja de execução, implementação ou de algoritmo. Vamos considerar dois casos:

- Encher a geladeira usando um portal
- Preparação de um bolo de fubá

No colocar agua na geladeira, podemos ter os seguintes processos:

(1) Abra um portal ligando a fonte de água ao reservatório na geladeira
(2) Encha o reservatório

Bem, temos um plano, embora seja BEM estranho, ainda sim é um algoritmo pois mostra passo a passo a se fazer. Contudo, temos uma enorme dificuldade com um passo: Como que abre um portal?

Embora alguns possamos fazer alguns planos, como no caso do portal, é impossível fazer isso, executar o plano. Este caso é um caso de erro de implementação, não se consegue aplicar na vida real. Existem outros problemas que ocorrem devido à implementação e ocorrem seja por recurso financeiro(exigir mais dinheiro que existe no mundo), físico(exigir que exista um portal), social(exigir que uma população faça alguma ação) e outros.

Já no caso da preparação de bolo cujo algoritmo está descrito abaixo, tem-se um erro de execução.

(1) Em um liquidificador adicione 3 ovos, 2 xicaras de açucar, 2 de fubá e 1 copo de leite.
(2) Bata bem até ficar homogêneo.
(3) Despeje a massa em uma forma untada e polvilhada
(4) Leve para assar em forno médio 180^oC pré-aquecido por 40 minutos.

Temos que se não temos um copo de leite, não poderemos continuar a executar o algorítmo. Esse erro é chamado de erro de execução pois durante o processo, não se há todas as ferramentas necessárias embora essas ferramentas sejam de fácil acesso.

Agora, existem os erros de algoritmo que são identificados quando não há erro de implementação nem de execução e a saída esperada não ocorre. Por exemplo, supomos que tenhamos o algorítmo que testa, começando por 1, de um em um para achar a raiz de um número. Assim, podemos achar as raizes de 4, 9, 16, 1764 e outros. Contudo, esse algorítmo não sabe calcular a raiz de 2,25 que é 1,5.

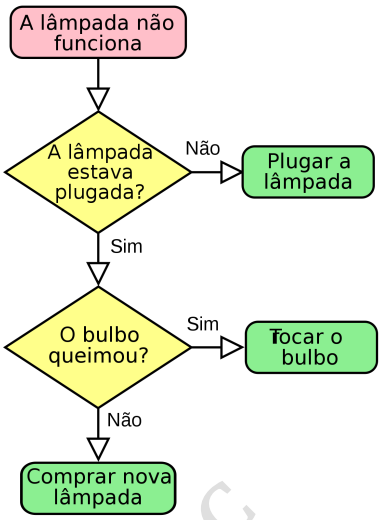
Examinar a solução. Uma vez que achamos uma solução(correta, sem erros), por exemplo, no caso de colocar a água na geladeira, podemos analisar se aquela é a melhor maneira de se fazer. Um dos primeiros passos a se fazer é verificar se o algoritmo funciona. Depois que o algoritmo funciona que se pensa em melhorar ou inventar outro algoritmo que seja melhor.

Projeto do algoritmo

Descrição narrativa. A descrição narrativa tenta transmitir o algoritmo através da fala, assim como feito a alguém que pede informação na rua, você indica(normalmente falando) em quais lugares virar ou seguir reto. Normalmente a descrição narrativa é suficiente para alguns casos que já se tem um bom conhecimento prévio, como reconhecer uma esquina, uma loja e outros. Contudo, a descrição narrativa também pode propagar erros como pode-se perceber no contar de histórias boca a boca.

Fluxograma. O fluxograma é uma outra maneira de exprimir algorítimos através de desenho de fácil compreensão. Um exemplo de fluxograma é mostrado pela Figura 1
Um fluxograma sempre tem um e somente um início, como no caso da Figura 1, tem-se "A lâmpada não funciona" que sempre possui setas saindo de si mas não entrando. Um fluxograma normalmente tem um fluxo de ações que é dado por suas setas. Em um fluxograma podem existir condicionais e ciclos.

FIGURE 1.



Pseudocódigo. Um pseudocódigo é similar à uma descrição narrativa, em que você utiliza de texto para descrever ações. Contudo, diferentemente dele, é composto por regras que se chama sintaxe, cujo exemplo do fluxograma é dado abaixo:

Algorithm 1 Consertar a lampada

```
1: se lampada_nao_esta_plugada então
2:     plugar_a_lampada
3: senão
4:     se bulbo_esta_queimado então
5:         trocar_o_bulbo
6:     senão
7:         comprar_nova_lampada
8:     fim se
9: fim se
```

Podemos ver que durante a implementação, utilizamos as palavras especiais ”se”, ”então”, ”senão” por exemplo. Essas são algumas regras de sintaxe. Por exemplo, a condicional ”se” só pode ser expressa pela palavra ”se” e não também pela palavra ”caso” como às vezes estamos acostumados. O computador prefere que falemos de uma determinada maneira com ele, de modo que entenda, assim como precisamos que alguém fale português para quem fale português. Essas palavras são palavras reservadas, outras semelhantes são mostradas como ”enquanto” e ”faça” no caso em que você deve descascar todas as batatas de um saco.

Algorithm 2 Exemplo do `enquanto`

```
1: enquanto existe_batata_no_saco faça
2:     retirar_batata_no_saco
3:     descascar_a_batata
4:     guardar_batata_descascada
5: fim enquanto
```

Falando superficialmente sobre a próxima aula. A implementação dos algoritmos normalmente se chama código, ou seja, se você implementa um algorítimo em C, você têm um código em C. Existem diversas linguagens de programação e para esse curso faremos com a linguagem C. Essa linguagem, bem como outras como Java e Python, possuí âmbito internacional e então não pode-se utilizar as palavras "se", "enquanto" entre outras. Assim, cada linguagem tem sua determinada sintaxe(ou palavras) para escrever e todos devem obedecer caso queira programar nessa linguagem. Consequência disso é que qualquer pessoa no mundo poderá utilizar seu código. Um exemplo de código em C está descrito abaixo

```
1  /*   Codigo: Exemplo de codigo em C
2       Autor: Carlos Adir
3  */
4  #include <stdio.h>
5  int main()
6  {
7    /* comentario */
8    int a = 0;
9    if(a == 0)
10     printf("a = 0, como esperado!\n");
11   else
12     printf("Deu algo errado!\n");
13   while (a < 3)
14   {
15     a++;
16     printf("a = %d\n", a);
17   }
18   return 0;
19 }
```

Por enquanto, não se preocupe com o código acima, será melhor explicado nas aulas posteriores detalhadamente! Só explicaremos que em um código em C é necessário 3 itens:

- Cabeçalho - Linhas 1-3
- Bibliotecas - Linha 4
- Código principal - Linhas 5-19

Suas utilidades serão apresentadas ao longo do curso. Mas podemos dizer que o cabeçalho é responsável por informar ao leitor(não somente você) sobre o que o código faz. O cabeçalho sempre será um comentário, enquanto as bibliotecas e código principal será melhor descrito ainda.