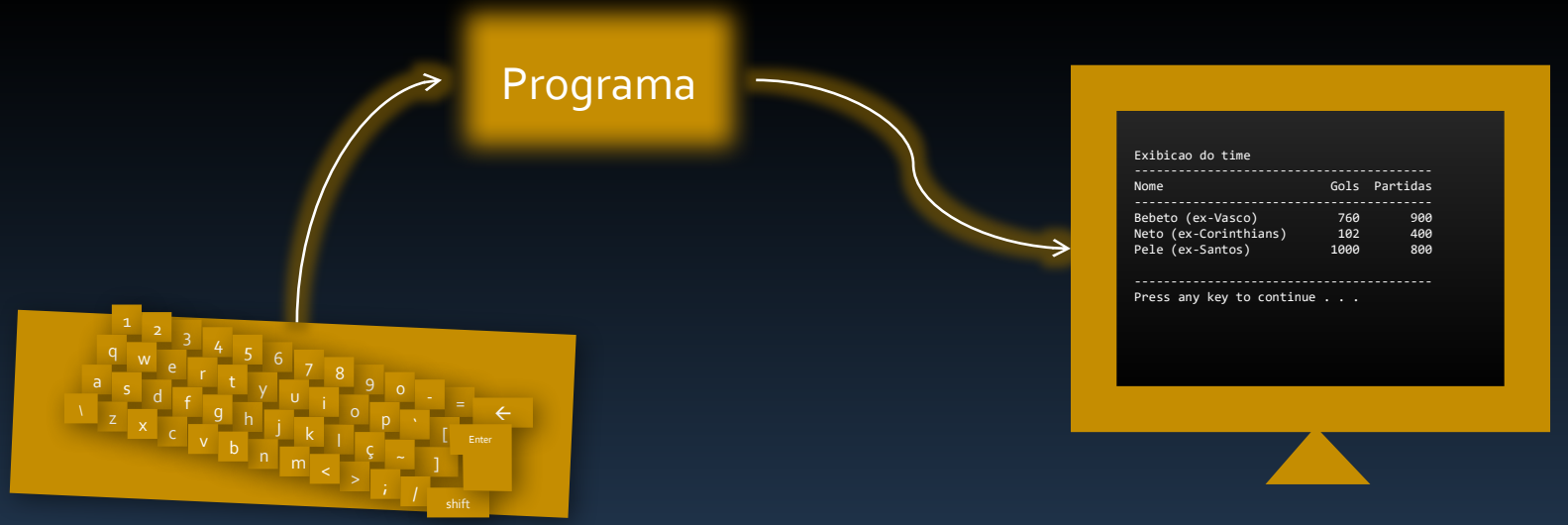
Programação de Computadores

ARQUIVOS TEXTO

Judson Santos Santiago

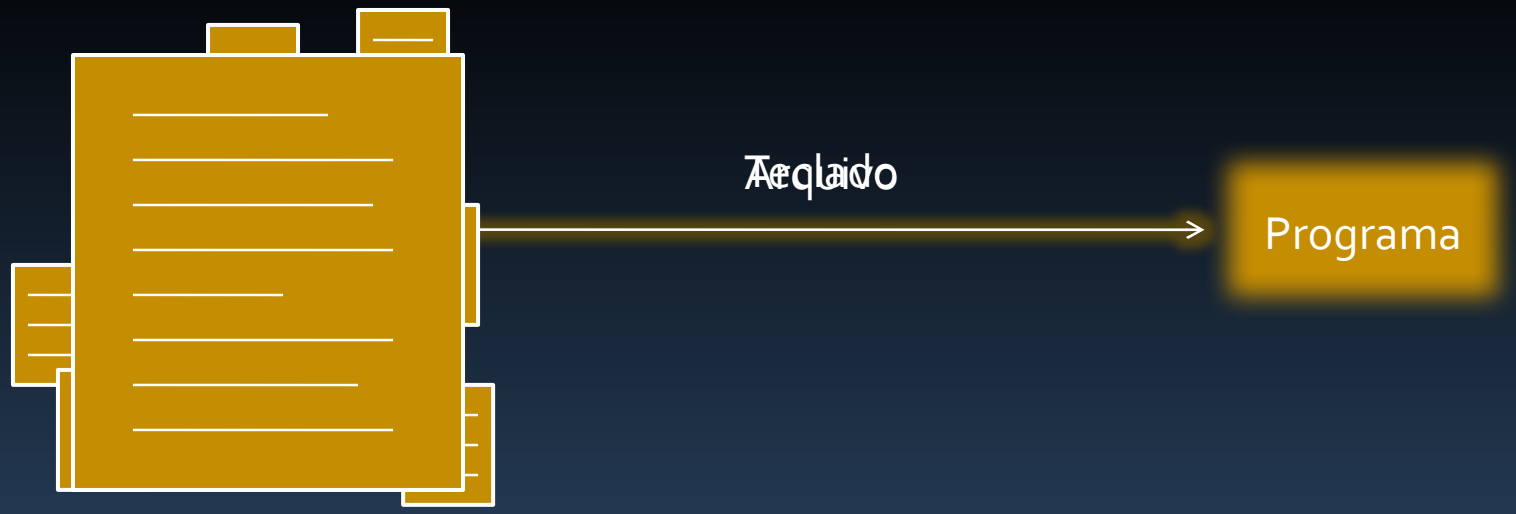
Introdução

- Frequentemente no computador a entrada de dados é feita pelo teclado e a saída de dados é feita em tela



Introdução

- A entrada de dados pelo teclado nem sempre é a melhor opção:
  - Imagine usar o teclado para entrar com nome, preço e quantidade em estoque de 1000 produtos



Introdução

- As vezes a saída de dados na tela não é a melhor opção:
  - Seria conveniente gerar uma lista de produtos fora de estoque em um arquivo

Introdução

- Os programas de computador trabalham com arquivos
  - Documentos, planilhas, apresentações, imagens, vídeos, sons, etc. são armazenados e manipulados a partir de arquivos
  - Compiladores lêem o arquivo fonte de um programa e geram um arquivo executável

- Um arquivo é um conjunto de bits guardado em algum dispositivo de armazenamento permanente
  - SSD, HDD, Pen-Drive, etc.

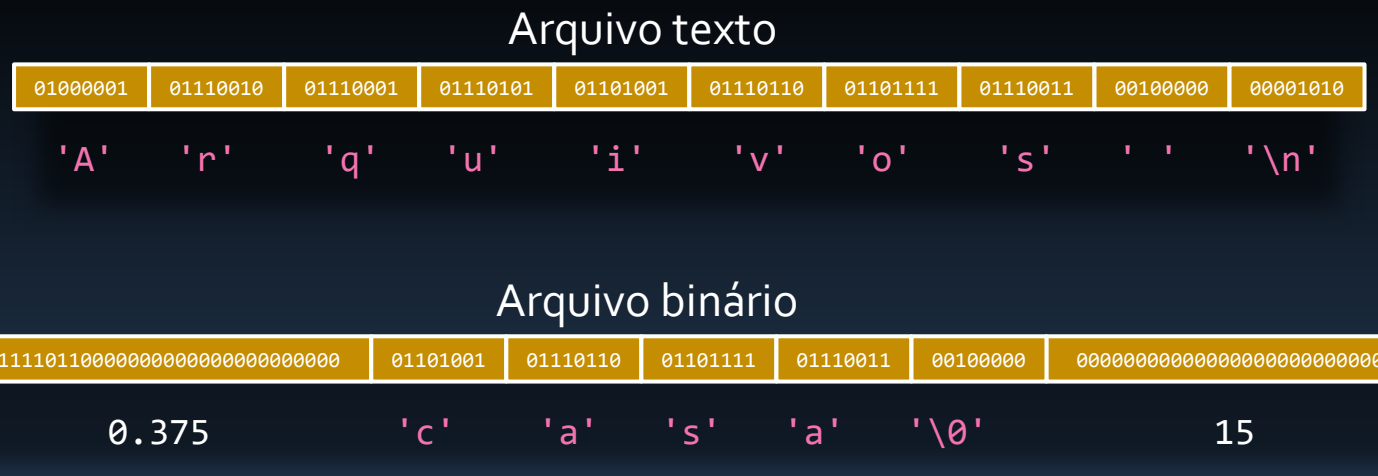
Introdução

- O sistema operacional se encarrega de gerenciar os arquivos
  - Saber a sua localização no disco
  - Guardar informações sobre o arquivo
    - Tamanho, datas de criação e modificação, permissões, etc.

- Como programador estamos interessados em conectar um programa a um arquivo para:
  - Ler informações do arquivo
  - Gravar informações no arquivo

Introdução

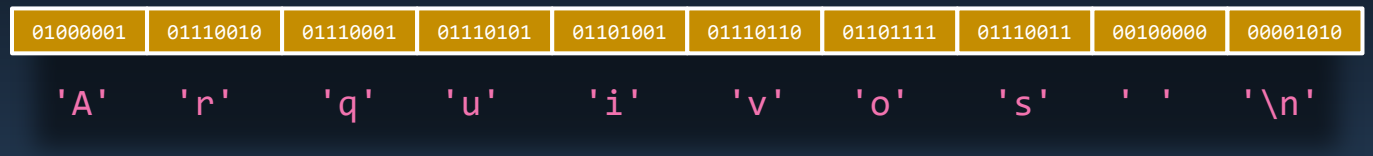
- Para um programador os arquivos se dividem em:
  - Arquivos texto
  - Arquivos binários



Introdução

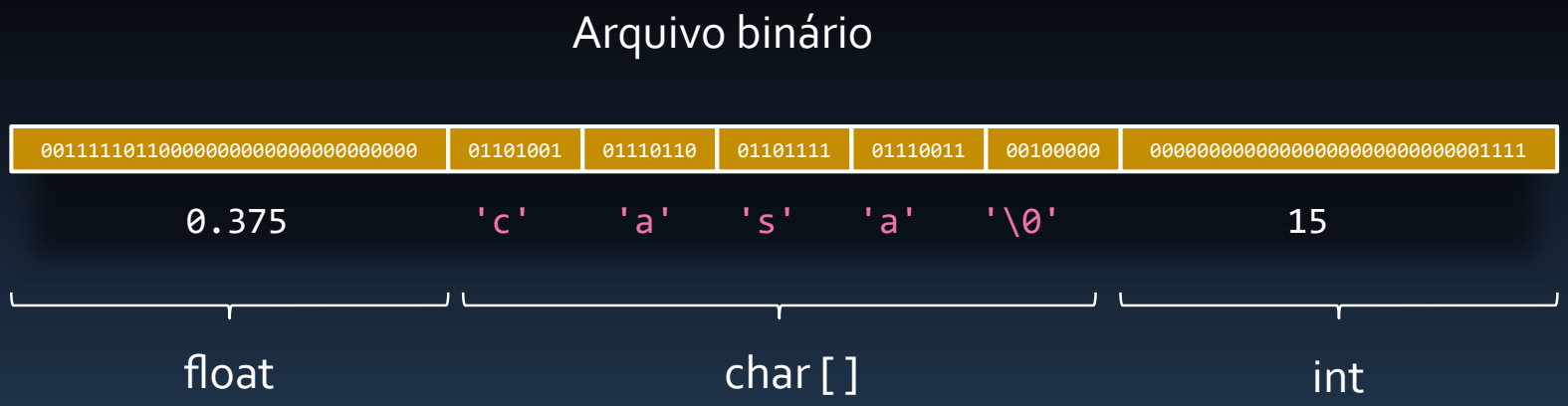
- Arquivos texto
  - Uma sequência de bits em que n bits representam um caractere
  - Na codificação ASCII um caracatere são 8 bits
  - Em Unicode um caractere tem de 8 a 32 bits
    - Existem várias codificações (UTF-8, UTF-16, UTF-32)

Arquivo texto na codificação ASCII



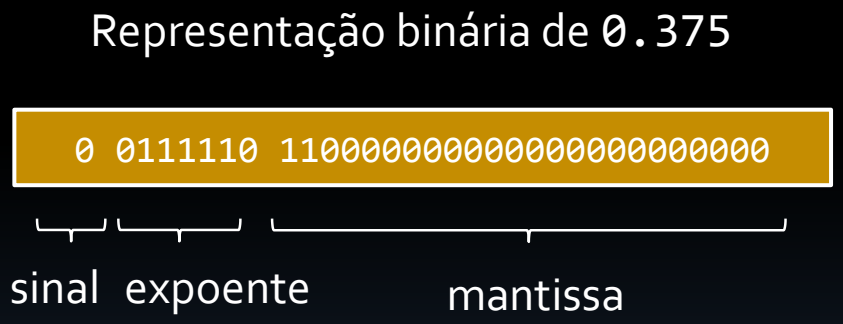
Introdução

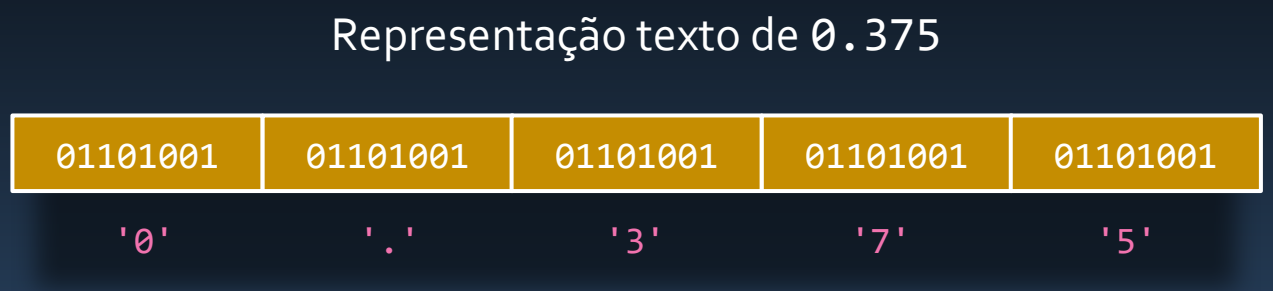
- Arquivos binários
  - Uma sequência de bits em que um conjunto de n bits representa uma informação na sua forma nativa (inteira, ponto-flutuante, caractere, etc.)



Introdução

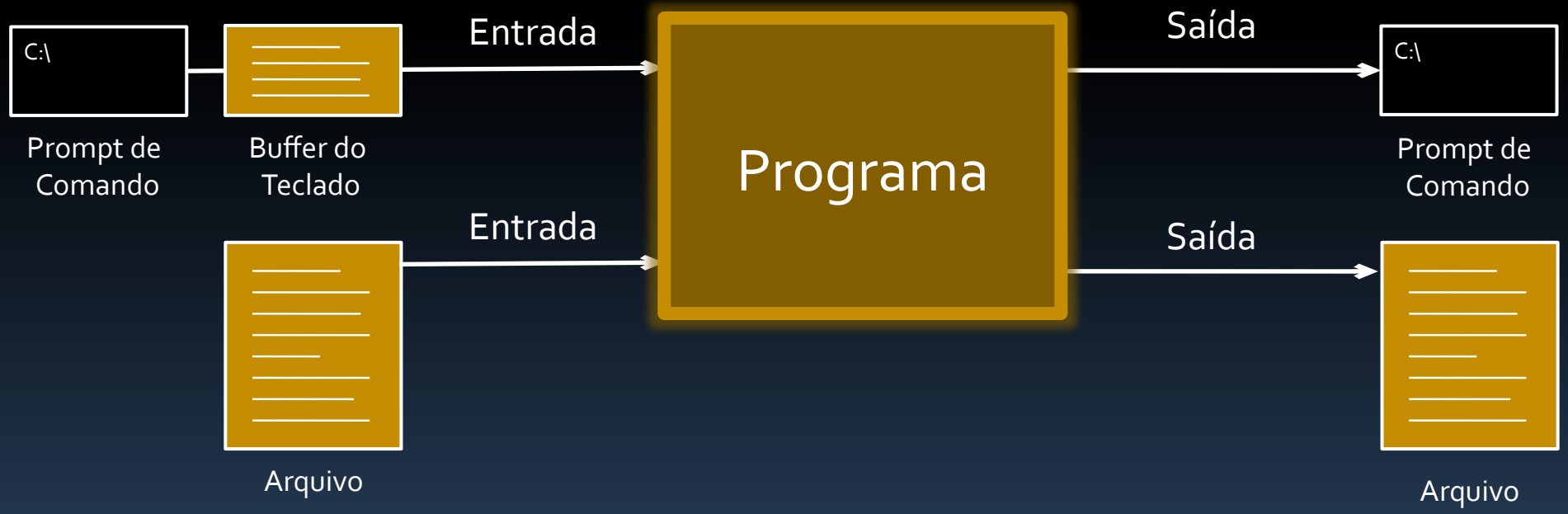
- Diferença entre arquivos texto e binário:





Arquivos Texto

- A entrada e saída em arquivos texto é muito parecida com a entrada e saída feita no terminal de comandos



Saída em Tela

- Para usar cout em um programa é necessário:
  - Incluir o arquivo de cabeçalho iostream
    - Define uma classe ostream para manipular a saída
    - Declara um objeto da classe ostream chamado cout

```
#include <iostream>
```

  - Considerar que os objetos estão definidos no espaço de nomes std

```
using namespace std;
// ou
using std::cout;
```

Saída em Tela

- Para usar cout em um programa é necessário:
  - Usar cout com o operador de inserção <<

```
char ch = 'M';
float num = 35.4f;
cout << ch;
cout << num;
```

  - Usar cout.setf() e cout.precision() para modificar exibição dos dados

```
float tax = 31.28062;
cout.setf(ios_base::fixed, ios_base::floatfield);
cout.precision(2);
cout << tax;
```

Saída em Arquivo Texto

- Fazer a saída em arquivos é semelhante:
  - Incluir o arquivo de inclusão fstream
    - Define uma classe ofstream para manipular a saída
    - Não existe um objeto predefinido
  - Criar um objeto do tipo ofstream
  - Associar um objeto ofstream com um arquivo do disco usando a função open()
  - Usar o operador de inserção << com o objeto ofstream
  - Fechar o arquivo com a função close()

Saída em Arquivo Texto

- O arquivo de inclusão fstream não fornece um objeto predefinido, é preciso criar um:

```
ofstream fout;                  // um objeto ofstream
```

- E associá-lo com um arquivo do disco:

```
fout.open("pesca.txt");       // fout associado ao arquivo pesca.txt
```

- Para usá-los desta forma:

```
double wt = 125.8;
char linha[81] = "objetos são variáveis de uma classe";
fout << wt << '\n';           // escreve um número
fout << linha << endl;        // escreve uma linha de texto
```

Saída em Arquivo Texto

- O objeto do tipo ofstream é uma variável comum e, portanto, pode ter um nome qualquer

```
ofstream saida;                 // um objeto ofstream
```

- O nome do arquivo pode ser obtido do usuário:

```
char arquivo[50];               // um vetor de caracteres
cin >> arquivo;                 // nome definido pelo usuário
saida.open(arquivo);            // abre arquivo definido pelo usuário
```

- E utilizado da mesma forma:

```
double wt = 125.8;
saida << wt;                    // escreve um número
```

Saída em Arquivo Texto

```
#include <iostream>
#include <fstream>
using namespace std;
int main()
{
     char carro[50];
     int ano;
     float precoN;                    // preço normal
     float precoP;                    // preço promocional

     ofstream fout;                   // cria objeto para saída
     fout.open("carinfo.txt");        // associa com arquivo

     cout << "Entre com a marca e modelo do carro: ";
     cin.getline(carro, 50);
     cout << "Entre com o ano: ";
     cin >> ano;
     cout << "Digite o preço normal: ";
     cin >> precoN;
```

// Promocional.cpp

Saída em Arquivo Texto

```
        precoP = 0.913f * precoN;

        // mostrando informações na tela
        cout.setf(ios_base::fixed, ios_base::floatfield);
        cout.precision(2);
        cout << "Marca e modelo: " << carro << endl;
        cout << "Ano: " << ano << endl;
        cout << "Preço normal: R$" << precoN << endl;
        cout << "Preço promocional: R$" << precoP << endl;

        // gravando informações no arquivo
        fout.setf(ios_base::fixed, ios_base::floatfield);
        fout.precision(2);
        fout << "Marca e modelo: " << carro << endl;
        fout << "Ano: " << ano << endl;
        fout << "Preço normal: R$" << precoN << endl;
        fout << "Preço promocional: R$" << precoP << endl;

        fout.close();
}
```

Saída em Arquivo Texto

- Saída do programa:

```
Entre com a marca e modelo do carro: Mustang GT
Entre com o ano: 2019
Digite o preço normal: 300000

Marca e modelo: Mustang GT
Ano: 2019
Preço normal: R$300000.00
Preço promocional: R$273900.00
```

- Arquivo carinfo.txt:

```
Marca e modelo: Mustang GT
Ano: 2019
Preço normal: R$300000.00
Preço promocional: R$273900.00
```

Saída em Arquivo Texto

- O método open():
  - Cria um novo arquivo se ele não existir
  - Sobrescreve o arquivo, se ele existir

```
fout.open("carinfo.txt");  // associa com arquivo
```

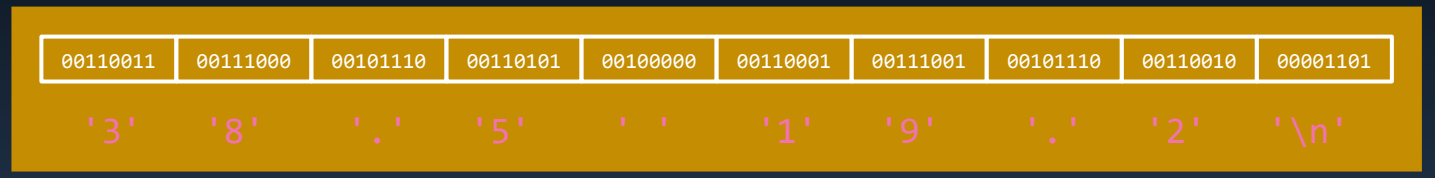
- A abertura de um arquivo pode falhar caso o arquivo utilizado seja um arquivo de acesso restrito
  - Um arquivo já aberto em outro programa
  - Um arquivo protegido pelo S.O.

Entrada pelo Teclado

- Quando utiliza-se `cin` para ler dados, o programa vê a entrada como uma série de caracteres (bytes):
  - Suponha que você digite a seguinte entrada:

    ```
    38.5 19.2
    ```

  - A entrada é armazenada no buffer de entrada como um conjunto de caracteres (1 caractere == 1 byte)

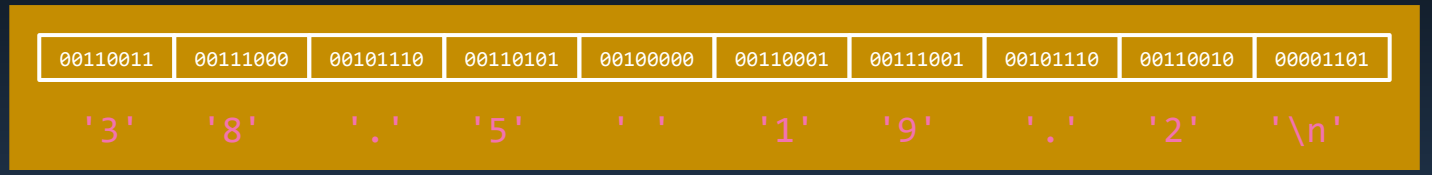


Entrada pelo Teclado

- Ao fazer a leitura de dados:
  - Cada byte é interpretado como um caractere
  - Não importa o tipo de destino, a entrada é sempre texto

```
cin >> total;
```

  - `cin` tem a responsabilidade de traduzir o texto para o tipo utilizado



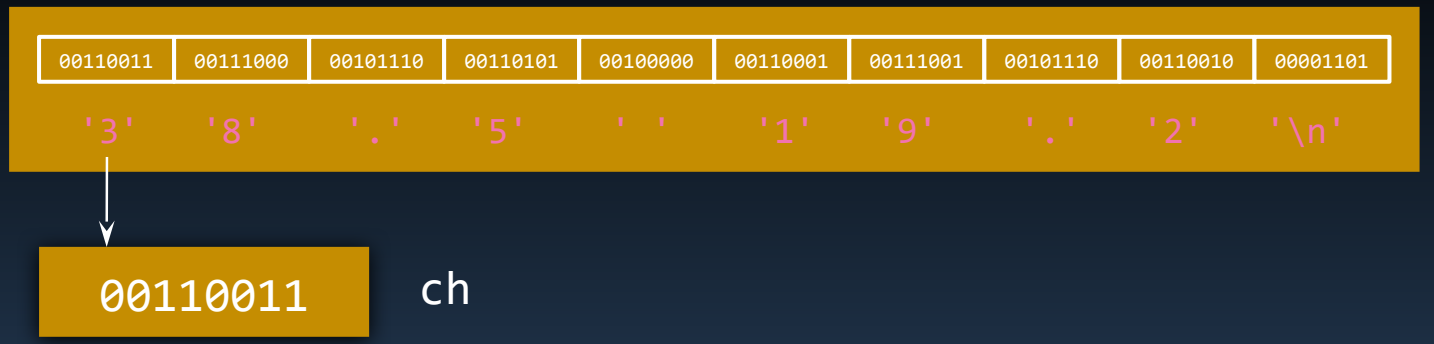
Entrada pelo Teclado

- Entrada usando uma variável tipo char:

```
38.5 19.2
```

- O primeiro caractere da entrada é atribuído para a variável, nenhuma tradução é necessária

```
char ch;
cin >> ch;
```



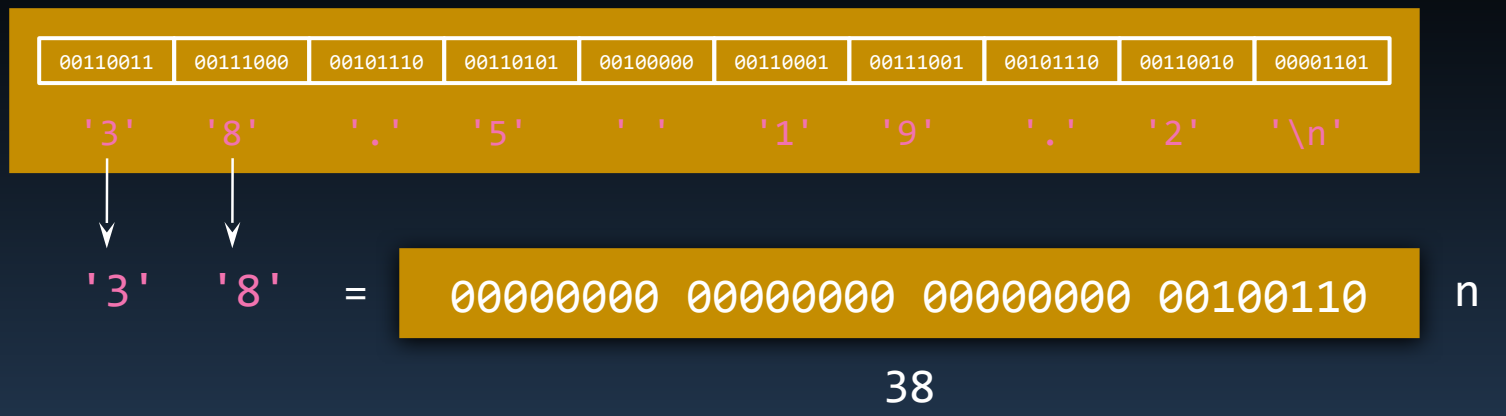
Entrada pelo Teclado

- Entrada usando uma variável tipo int:

```
38.5 19.2
```

- A leitura é feita até encontrar o primeiro caractere que não é um dígito

```
int n;
cin >> n;
```



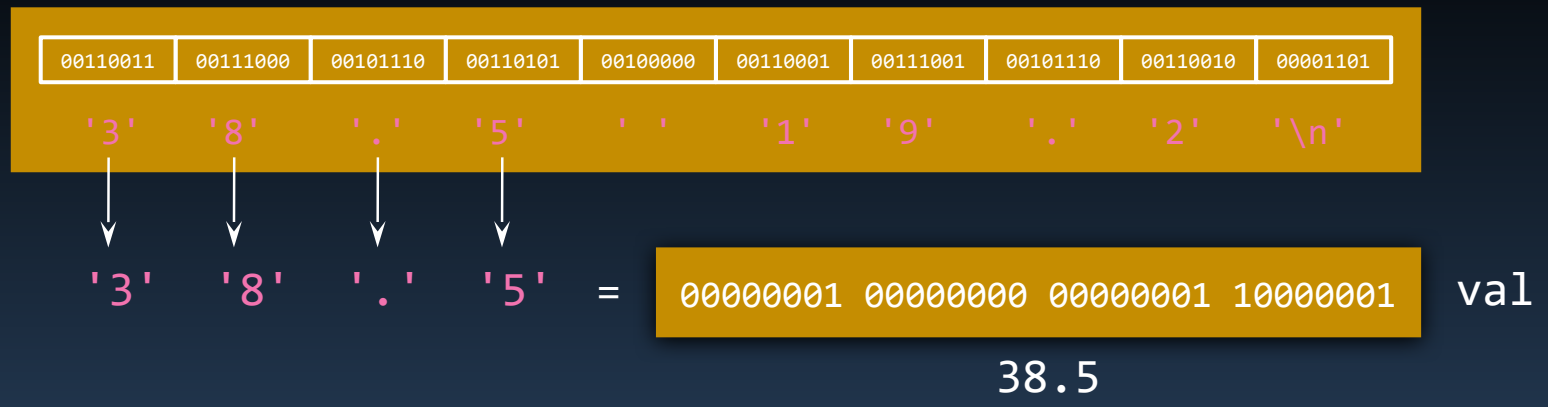
Entrada pelo Teclado

- Entrada usando uma variável tipo float:

```
38.5 19.2
```

- A leitura é feita até encontrar um caractere que não faz parte de um ponto-flutuante

```
float val;
cin >> val;
```



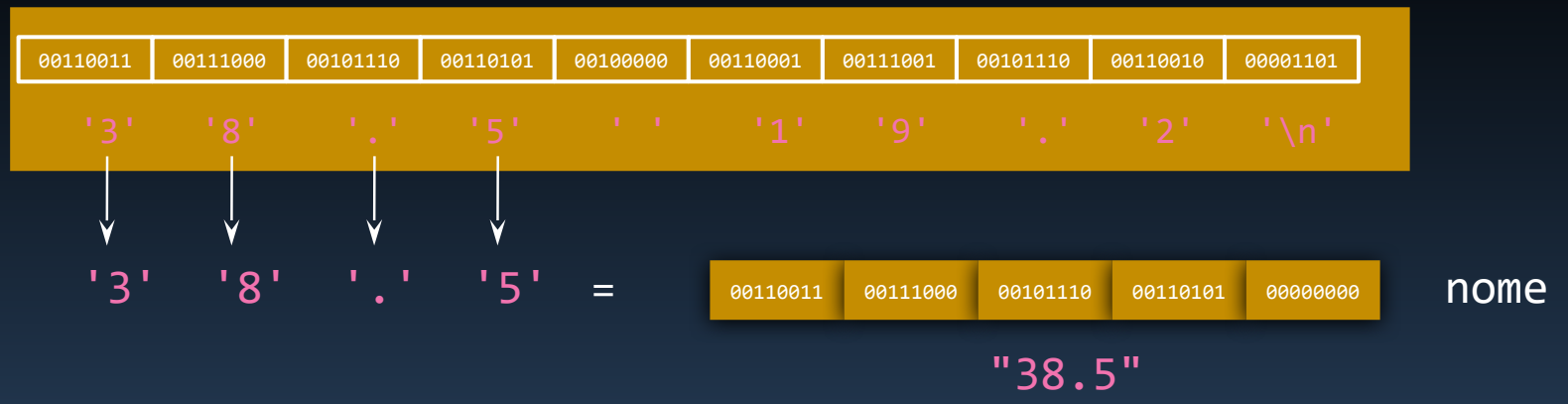
Entrada pelo Teclado

- Entrada usando um vetor de caracteres:

```
38.5 19.2
```

- A leitura é feita até encontrar um caractere em branco, tabulação ou marcação de nova linha

```
char nome[50];
cin >> nome;
```



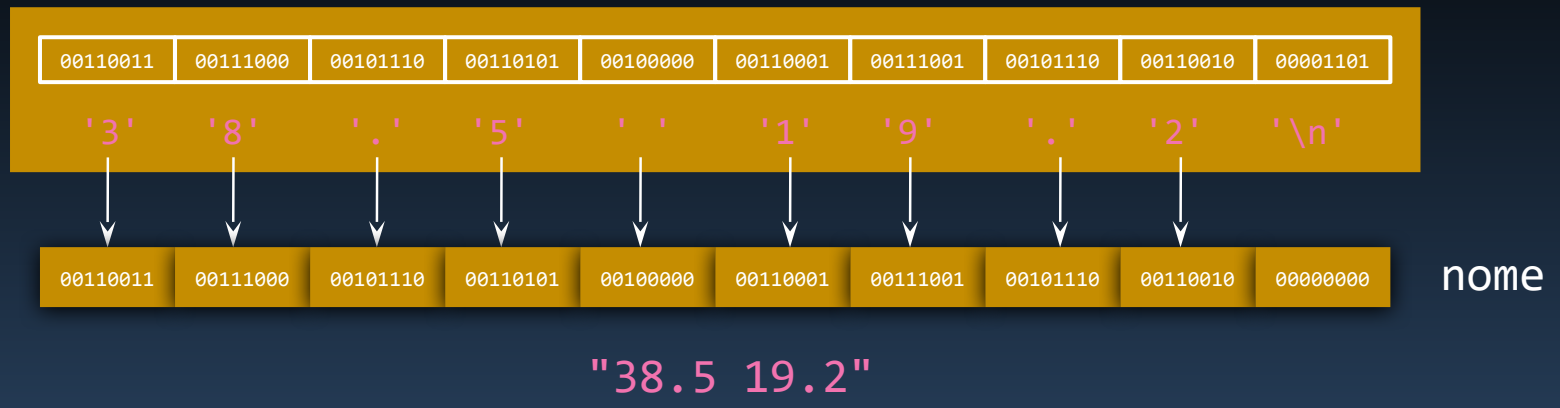
Entrada pelo Teclado

- Entrada usando um vetor de caracteres:

```
38.5 19.2
```

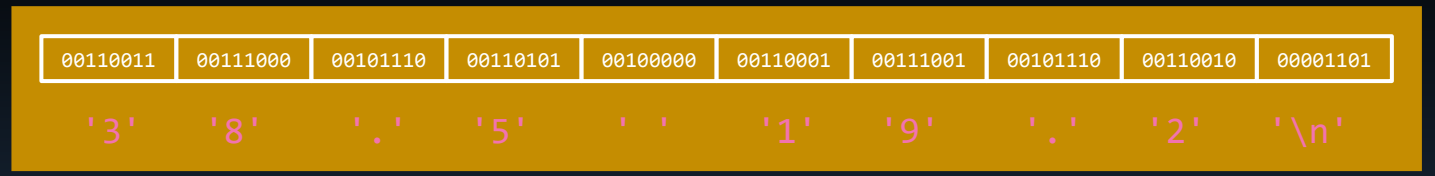
- Com getline, a entrada é lida até o caractere de nova linha

```
char nome[50];
cin.getline(nome, 50);
```



Entrada pelo Teclado

- Em resumo, a entrada pelo teclado é recebida sempre em forma de texto e depois convertida para o tipo apropriado
  - Um arquivo texto se assemelha muito ao buffer do teclado, isto é, ele é composto por uma seqüência de caracteres



  - Um arquivo binário armazena mais que texto

Entrada pelo Teclado

- Para usar `cin` em um programa é necessário:
  - Incluir o arquivo de inclusão iostream
    - Define uma classe istream para manipular a entrada
    - Declara um objeto da classe istream chamado cin

```
#include <iostream>
```

  - Considerar que estes objetos estão definidos no espaço de nomes std

```
using namespace std;
//ou
using std::cin;
```

Entrada pelo Teclado

- Para usar `cin` num programa é necessário:
  - Usar cin com o operador de extração >>

```
char ch;
float num;
cin >> ch;
cin >> num;
```

  - Usar cin.get() para ler caracteres ou cin.getline() para ler uma linha

```
char ch;
char nome[30];
cin.get(ch);
cin.getline(nome, 30);
```

Leitura de Arquivos Texto

- A leitura de arquivos é muito semelhante:
  - Incluir o arquivo de cabeçalho fstream
    - Define uma classe ifstream para manipular a entrada
    - Não existe um objeto predefinido
  - Criar um objeto do tipo ifstream
  - Associar um objeto ifstream com um arquivo do disco usando a função open()
  - Usar o operador de extração >>, o método get() ou getline()
  - Fechar o arquivo com a função close()

Leitura de Arquivos Texto

- O arquivo de inclusão fstream não fornece um objeto predefinido, é preciso criar um:

```
ifstream fin;                    // um objeto ifstream
```

- E associá-lo com um arquivo do disco:

```
fin.open("boliche.txt");         // fin associado ao arquivo boliche.txt
```

- Para usá-lo desta forma:

```
double wt;
char linha[81];
fin >> wt;                       // lê um número de boliche.txt
fin.getline(linha, 81);          // lê uma linha de texto
```

Leitura de Arquivos Texto

- O objeto do tipo ifstream é uma variável comum e, portanto, pode ter um nome qualquer

```
ifstream entrada;                    // um objeto ifstream
```

- O nome do arquivo pode ser obtido do usuário:

```
char arquivo[50];                    // um vetor de caracteres
cin >> arquivo;                      // nome definido pelo usuário
entrada.open(arquivo);               // abre arquivo definido pelo usuário
```

- E utilizado da mesma forma:

```
double wt;
entrada >> wt;                       // lê um número do arquivo
```

Leitura de Arquivos Texto

- Se o arquivo não existir no disco, todas as tentativas de uso do objeto ifstream vão falhar
  - A melhor forma de verificar se um arquivo foi aberto corretamente é através do método is_open()

```
fin.open("boliche.txt");
if (!fin.is_open())
{
    cout << "A abertura do arquivo falhou!" << endl;
    exit(EXIT_FAILURE);
}
```

Leitura de Arquivos Texto

```cpp
#include <iostream>
#include <fstream>
using namespace std;
const int Tam = 60;
int main()
{
     char arquivo[Tam];
     ifstream fin;                    // cria objeto para leitura de arquivo

     cout << "Digite nome do arquivo: ";
     cin.getline(arquivo, Tam);
     fin.open(arquivo);               // associa com arquivo do disco

     if (!fin.is_open())              // a abertura do arquivo falhou
     {
        cout << "A abertura do arquivo " << arquivo << " falhou!" << endl;
        cout << "Programa encerrando.\n";
        exit(EXIT_FAILURE);
     }
```

// MediaArq.cpp

Leitura de Arquivos Texto

```
double valor;                   // valor lido do usuário
double soma = 0.0;              // soma dos itens
int cont = 0;                   // número de itens lidos

fin >> valor;                   // lê primeiro valor
while (fin.good())              // enquanto a entrada for boa e não EOF
{
   ++cont;                      // mais um item lido
   soma += valor;               // acumula valores lidos
   fin >> valor;                // lê próximo valor
}

if (fin.eof())
   cout << "Fim de arquivo alcançado.\n";
else if (fin.fail())
   cout << "Tipo incorreto de dado na entrada.\n";
else
   cout << "Entrada encerrada por razão desconhecida.\n";
```

Leitura de Arquivos Texto

```
    if (cont == 0)
       cout << "Nenhum valor lido.\n";
    else
    {
       cout << "Itens lidos: " << cont << endl;
       cout << "Soma:  " << soma << endl;
       cout << "Média: " << soma / cont << endl;
    }

    fin.close();    // fecha o arquivo

    return 0;
}
```

Leitura de Arquivos Texto

- Arquivo "pontos.txt":

```
18 19 18.5 13.5 14
16 19.5 20 18 12 18.5
17.5
```

- Saída do programa:

```
Digite nome do arquivo: pontos.txt
Fim de arquivo alcançado.
Itens lidos: 12
Soma: 204.5
Média: 17.0417
```

Resumo

- Entrada e saída em arquivos é um recurso importante
  - Utilizado por praticamente todos os programas
- Os dados podem ser armazenados em:
  - Arquivos texto: podem ser lidos por qualquer editor de texto
  - Arquivos binários: são mais precisos para armazenar números ponto-flutuante, as operações de leitura e escrita são mais rápidas, mas geralmente ocupam mais espaço