***4.1.0 Apresentação dos Dados Coletados e Análises Realizadas***

Nesta seção, serão apresentados os dados coletados a partir da pesquisa bibliográfica e da análise documental, seguidos de uma interpretação dos resultados em relação ao contexto histórico e científico que possibilitou o surgimento do computador moderno e a Análise dos limites suportados pelos tipos de arquivos. A metodologia adotada permitiu uma análise crítica sobre como as técnicas computacionais desenvolvidas no passado influenciaram diretamente as tecnologias modernas.

***4.1.1 Análise dos limites suportados pelos tipos de arquivos***Os limites numéricos que cada tipo inteiro estão definidos na biblioteca da linguagem LIMITS.H, na forma de constantes.de acordo com a capacidade de cada variável foram ultilizadas as variaveis Char com a capacidade de 1 byte, valor mínimo de -128 e valor máximo de 127. Unsigned char 1 byte, valor mínimo de 0 e valor máximo de 255. Short int 2 bytes, valor mínimo -32768 e 32767. Unsigned short int 2 bytes, valor mínimo 0 e valor máximo 65535. Int 4 bytes, valor mínimo de -2147483648 e valor máximo de 2147483647. Unsigned int 4 bytes, valor mínimo de 0 e valor máximo de 4294967295. Long int 4 ou 8 bytes valor mínimo de -2147483648 e máximo de 2147483647. Unsigned long int 4 ou 8 bytes valor mínimo de 0 e máximo de 4294967295. Long long int 8 bytes valor mínimo de -9223372036854775808 e valor máximo de 9223372036854775807. Unsigned long long int 8 bytes, valor mínimo de 0 e valor máximo de 18446744073709551615.   
Obs.: long int = 4 bytes (32 bits) ou 8 bytes (64 bits); unsigned long int = 4 bytes (32 bits) ou 8 bytes (64 bits).  
  
***4.1.2 o que acontece se você utilizar um valor fora dos limites dos tipos de variáveis.*Overflow e Underflow**:  
Overflow ocorre quando um valor excede o limite máximo que uma variável pode armazenar.  
Underflow ocorre quando o valor é menor que o limite mínimo que a variável pode armazenar.

O que pode acontecer:  
Linguagens com comportamento de rotação (como C e C++):

Em linguagens como C e C++, o overflow e underflow podem resultar em “comportamento de rotação” (wrap-around). Isso significa que, quando o valor excede o limite máximo, ele “volta” ao limite mínimo e vice-versa. Isso pode causar erros difíceis de detectar.

Perda de precisão:

Para variáveis de ponto flutuante (como float e double), valores fora dos limites podem resultar em infinito positivo ou negativo (Infinity, -Infinity), ou em NaN (Not a Number). Além disso, operações com valores muito pequenos podem resultar em perda de precisão devido ao underflow.

Erro de compilação:

Em algumas linguagens mais seguras ou em configurações mais restritivas, tentar atribuir um valor fora do limite de uma variável pode resultar em um erro de compilação.

Exemplos:

C/C++: Se você declarar um unsigned int (sem sinal) e atribuir um valor negativo, isso resultará em um overflow, onde o número será convertido para um grande número positivo.

Esses cenários ilustram a importância de conhecer os limites de cada tipo de dado em uma linguagem.  
  
***4.1.3 Função que invalida alguns caracteres***Uma função em C que invalida caracteres geralmente percorre uma string e altera ou remove certos caracteres indesejados. Isso pode ser feito de diferentes maneiras, dependendo de como você define "invalidação" que no caso ficou definida por “0”. Aqui estão algumas abordagens comuns:

Substituir caracteres inválidos por outro caractere

Um exemplo clássico é substituir caracteres inválidos (como caracteres não alfabéticos) por um caractere específico, como “0” (nulo).   
  
Remover caracteres inválidos (compactação da string)

Outra forma de invalidar caracteres é removê-los completamente da string, "compactando" a string para que os caracteres válidos permaneçam.  
  
**Explicação Geral de Como Funciona**:  
  
1. Percorrer a string: A função percorre a string usando um loop que verifica cada caractere até encontrar o caractere nulo “0”, que marca o fim da string.

2. Verificar caracteres: Para verificar se um caractere é válido ou inválido, usamos funções auxiliares da biblioteca <ctype.h>, como isalpha (para letras) ou isdigit (para números).

3. Substituir ou remover: Dependendo do que você deseja fazer com os caracteres inválidos, você pode:

- Substituí-los por outro caractere (como '0').

- Compactar a string, movendo os caracteres válidos para as primeiras posições e eliminando os inválidos.

4. Modificar a string original: Como as strings em C são apenas arrays de caracteres, a modificação é feita diretamente na string original.

Essas abordagens são úteis em situações como validação de entrada de usuário, limpeza de dados, ou formatação de strings antes de processamento posterior.  
  
***4.1.4 Função principal***  
  
Neste algoritmo, recebemos uma cadeia de caracteres em formato hexadecimal, e o primeiro passo é separar essa cadeia em pares de caracteres (duplas). Em seguida, aplicamos uma função de invalidação de caracteres para filtrar ou remover valores inválidos, determinando quais devem ser traduzidos.

Após a filtragem, convertemos as duplas de caracteres hexadecimais para seus valores correspondentes em decimal. Esses valores decimais são então mapeados para seus respectivos caracteres da tabela ASCII \*CP437\*.

Finalmente, a sequência de caracteres resultante é utilizada para imprimir a imagem na tela.  
  
***4.2.0 Contexto Histórico***

A análise bibliográfica revela que as bases para o desenvolvimento do computador moderno remontam ao século XVII, com a criação da Pascalina, a primeira calculadora mecânica funcional. A contribuição de Charles Babbage e Ada Lovelace no século XIX também foi crucial para o desenvolvimento dos conceitos de computação programável.

Contudo, foi no século XX, com o contexto da Segunda Guerra Mundial, que ocorreram avanços decisivos. A necessidade de realizar cálculos rápidos e precisos, particularmente no campo militar, impulsionou o desenvolvimento de máquinas como o ENIAC (Electronic Numerical Integrator and Computer), em 1946, considerado o marco inicial da era da computação moderna. O ENIAC foi essencial para cálculos balísticos e previsões meteorológicas, demonstrando a relevância prática da computação no contexto militar e científico.

O estudo também destacou a influência de Alan Turing, que, com sua Máquina de Turing Universal, idealizou o conceito de uma máquina programável capaz de executar qualquer algoritmo. Isso abriu caminho para a programação moderna e consolidou as bases teóricas da ciência da computação.  
  
***4.2.1 Evolução Científica e Tecnológica***

*Os dados históricos demonstram que a evolução da computação ocorreu em etapas, cada uma marcada por inovações tecnológicas que aumentaram a capacidade de processamento das máquinas. Durante a Segunda Guerra Mundial, máquinas como o Colossus foram desenvolvidas para decifrar códigos* secretos, marcando um ponto chave na história da criptografia e da computação.

A introdução dos transistores na década de 1950 e o desenvolvimento dos circuitos integrados na década de 1960 revolucionaram ainda mais a computação, tornando os computadores menores, mais rápidos e acessíveis. A análise documental de algoritmos e técnicas usados durante esse período mostrou como conceitos de automação e aritmética digital foram integrados em operações militares e científicas, exemplificados pelos cálculos balísticos realizados pelo ENIAC.  
  
***-- como os primeiros computadores enfrentaram limitações de hardware e software, particularmente em relação à manipulação de dados numéricos e ao sistema de codificação.***

***4.2.2 Limitações de Hardware***

Os primeiros computadores, como o ENIAC e o Colossus, apresentavam severas limitações de hardware:

Capacidade de armazenamento: A memória disponível era extremamente limitada, forçando os engenheiros a simplificar algoritmos e dividir o processamento em pequenos blocos de dados.

Velocidade de processamento: Embora inovadores para a época, a velocidade de cálculo desses computadores era muito baixa em comparação com as máquinas modernas, o que impactava diretamente a eficiência em processos como a decodificação de mensagens criptografadas.

Precisão e erros: O uso de números muito grandes ou pequenos podia gerar erros, pois esses computadores tinham uma capacidade limitada para representar frações ou decimais.

Sistemas físicos delicados: Componentes como válvulas e relés frequentemente apresentavam falhas, afetando a confiabilidade e a continuidade do processamento.

***4.2.3Limitações de Software***

Além das restrições de hardware, os primeiros computadores enfrentavam desafios relacionados ao software:

Programação manual: As instruções eram inseridas manualmente por meio de cartões perfurados, um processo demorado e propenso a erros.

Codificação rígida: Os algoritmos eram limitados a operações matemáticas básicas, complicando a execução de processos mais complexos, como a criptografia.

Falta de abstração: Não havia linguagens de alto nível ou compiladores. Todas as operações tinham que ser traduzidas diretamente em instruções binárias, tornando a programação mais desafiadora.  
  
***-- melhor compreensão sobre contexto histórico referente a decodificação de mensagens durante a Segunda Guerra Mundial que envolvia algoritmos que manipulavam grandes quantidades de dados, incluindo caracteres alfanuméricos e de tal forma que era comum remover ou invalidar certos caracteres ou informações desnecessárias para otimizar o processo de criptografia/decodificação.***

***4.3 Manipulação de Dados Numéricos e Limitações Computacionais***

Nos primeiros computadores, os algoritmos de criptografia e decodificação envolviam operações matemáticas complexas, como somas e multiplicações de grandes números, utilizando sistemas binários e decimais. As limitações de hardware e software obrigavam os operadores a lidar com restrições severas de memória e processamento.

A máquina Enigma, utilizada pelos alemães durante a guerra, utilizava padrões numéricos e alfanuméricos para criptografar mensagens. A decodificação dependia da capacidade de manipular grandes quantidades de dados de forma eficiente. As limitações tecnológicas tornavam esse processo mais lento, exigindo otimizações e avanços nos algoritmos de decodificação.

***-- paralelo direto com os processos de criptografia e decodificação de mensagens que foram fundamentais durante a Segunda Guerra Mundial.  
  
4.4 Análise dos Resultados e Comparação com a Segunda Guerra Mundial***

As análises mostram como as limitações tecnológicas da época exigiam soluções criativas para decodificar mensagens com rapidez e precisão. Isso pode ser comparado ao famoso caso da quebra da máquina Enigma pelos britânicos, que combinaram técnicas matemáticas com inovação tecnológica para superar os desafios de criptografia.

Da mesma forma, as dificuldades enfrentadas nos exercícios propostos revelam os desafios encontrados pelos primeiros computadores para processar grandes quantidades de dados. As técnicas de decodificação desenvolvidas durante a guerra, como o uso de padrões numéricos e a repetição de mensagens, podem ser vistas nas abordagens utilizadas para otimizar os processos computacionais da época.  
  
***-- Conclusão***

***4.5 Conclusão da Análise***

Os dados mostram que o desenvolvimento do computador moderno foi diretamente influenciado pelo contexto da Segunda Guerra Mundial, mas suas raízes remontam a séculos anteriores de evolução tecnológica. A análise documental e bibliográfica destacou que a computação se desenvolveu de forma incremental, com avanços que revolucionaram a ciência, a indústria e a sociedade.

As técnicas desenvolvidas na época, particularmente em relação à automação de cálculos e à criação de algoritmos programáveis, formam a base da computação moderna*.*

***4.6 Discussão***

A análise dos dados coletados oferece uma compreensão mais profunda sobre como a computação evoluiu ao longo do tempo. O desenvolvimento do computador foi impulsionado por necessidades práticas, como as operações militares da Segunda Guerra Mundial, e consolidado por avanços científicos.

***4.7 Implicações Futuras***

Os resultados da análise histórica sugerem que a computação está em constante evolução. Desde cálculos simples até algoritmos complexos e inteligência artificial, o futuro da computação aponta para inovações como os computadores quânticos e a inteligência artificial avançada. Essas tecnologias têm o potencial de revolucionar a forma como interagimos com a tecnologia, moldando o futuro da ciência e da sociedade