LISTA DE EXERCÍCIOS 6
ALUNO:GUSTAVO SANTOS TEIXEIRA
Questão 1000

```
#include <stdio.h>
#include <stdlib.h>
int main(){
    printf("Hello World");
  return 0;
}
```

Questão 1002

```
#include <stdio.h>
#include<stdlib.h>
#define pi 3,14159
int main(){
  float raio,area;
  printf("Informe o valor do raio : ");
  scanf("%f",&raio);
  area = ((raio^2)*pi);
  printf("O valor da área é %f",area);
  return 0;
}
```

Questão 1003

```
#include <stdio.h>
#include<stdlib.h>
int main(){
 int a,b,soma;
  printf("Informe dois valores inteiro : ");
  scanf("%d%d",&a,&b);
  soma = a+b;
  printf("SOMA = %d",soma);
  return 0;
}
```

Questão 1004

```
#include <stdio.h>
#include<stdlib.h>
int main(){
 int a,b,prod;
  printf("Informe dois valores inteiro : ");
  scanf("%d%d",&a,&b);
  prod = a*b;
  printf("PROD = %d",prod);
  return 0;
}
```

Anotações referentes a Linguagem C

- [Anotações sobre a linguagem C]
- Características:
  - Alto nível.
  - Possue algumas características de baixo nível.
  - Linguagem estruturada.
  - Possue proposito geral.
  - multiplataforma.
  - Ela foi uma inspiração para outras.
  - O Windows foi feito em C. (Sistemas operacionais).
- Histórico da Linguagem C:
  - BCPL (Basic Combined Programming Language)
    - (1966)
  - B (1969) - Simplificação

- UNIX - (Relacionado a Assembly).
- NBC (1972), mais tarde foi ~~rebat~~ batizado para C.
- Busca da "entrada/saída" funcionar em qualquer computador.
- Standard I/O (stdio) => (Entrada/Saída), biblioteca.
- (C++) > C mais orientação a objeto.

- Como funciona a linguagem C:
  - O computador entende sequências de bits (0 e 1).
  - Compilação = Tradução.
  - Fonte => Objeto => máquina.
  - Compilação => Código fonte para código objeto.
  - Código objeto é de baixo nível.
  - Montagem => Código objeto para código de máquina.
  - Compiladores específicos para cada tipo de computador.
  - Um mesmo código fonte pode ser compilado de maneira diferente e gerar resultados diferentes.
- História da linguagem C:
  - Criada em 1972 e implementada por Dennis Ritchie.
  - Derivada das linguagens Algol 68 e BCPL.
  - Foco => Desenvolvimento de sistemas operacionais e compiladores.
  - Usada na criação o Sistema UNIX.
  - Linguagem de propósito geral.
  - (C++) > 1986.
  - ANSI C (1999).
  - Houve uma busca para a padronização do C.
- Sistemas operacionais feitos em C:
  - ~~Glo~~ Google Chrome OS.
  - Android.
  - Windows NT e CE.

- Unix
- IOS
- Solaris
- HP-UX 11i

- Sintaxi e semântica em C:
  - Linguagem: Conjuto de regras sintáticas e semânticas usadas para definir uma forma de comunicação.
  - Sintaxe > Forma como é escrita, um exemplo são as chaves em C.
  - A semântica é complementar a sintaxe.
  - Semântica > Significado de instruções válidas de uma linguagem, basicamente oque as instruções fazem.
  - Semântica > Como se fosse a descrição da execução do programa em uma linguagem humana.
  - Semantica Estática:
    - Descreve a característica de um programa válido.
    - Geralmente declara as restrições de tipos.
  - Semântica Dinâmica:
    - Resultados da execução de um programa.
    - Operacional, denotacional e axiomática.
    - Semantica Operacional:
      - Significado de uma construção em uma máquina hipotética.
    - Semântica denotacional:
      - Define a semantica dos de linguagem de programação.
    - Semântica axiomática:
      - Propriedade do efeito da execução de estruturas.
- Processo de compilação:

- Compilação => Pega a saída do pré-processador e produz o arquivo objeto.
- Vinculação => Pega os arquivos objetos e produz uma biblioteca ou executável.
- A compilação é a única realizada na saída de cada pré-processador.
- Arquivos objetos podem ser colocados em bibliotecas.

- IDE:
  - Ambiente de desenvolvimento integrado (IDE),
  - Interface gráfica de usuário (GUI) que combina ferramentas comuns para desenvolvimento de ~~aplicação~~ aplicações.
  - Editor de código fonte:
    - Destaque da sintaxe com indicações visuais.
    - Preenchimento automático.
    - Verificação de bugs.
  - Automação de compilação local:
    - Automatizar tarefas simples e repetitivas.
    - Compilação de código fonte e código binário.
    - Criação de pacotes de códigos binários.
    - Execução de testes automáticos.
  - Debugger:
    - Testa o programa e mostra a localização do bug no código original.
- Melhor IDE para C:
  - Code Blocks / Dev-C++
- Linguagens produzidas a partir do C:
  - Java
  - Shell

- Javascript

- Compiladores:
  - O compilador processa e traduz para linguagem de máquina.
  - Analiza o código fonte, coleta e reorganiza suas instruções.
  - Um interpretador analiza o código linha por linha em sequência, sem olhar todo o programa.
  - Compilador exigem a criação de um executável.
  - Saída da compilação => Código objeto.

- Esqueleto de um programa em C:

```
#include <stdio.h>          -> Escrita e leitura.
#include <stdlib.h>         -> Biblioteca auxiliar.
int main() {                -> Quebra de linha.
    printf("Hello world!\n"); -> Escrever na tela.
    system("pause");
    return 0;               -> Retorna que está tudo OK.
}
```

  - Todo comando termina com ponto e vírgula.

- Declaração de variáveis:
  - É necessário ~~se~~ esclarecer o tipo e o nome.
  - char => Caracteres.
  - int => Inteiros.
  - float => números reais.
  - double => Número real com maior precisão.
  - Nome da variável:
    - maiúsculas e minúsculas.
    - números
    - Underscore (_ = ")

- não usar letras especiais e palavras reservadas.
- não é permitido numero no início de variáveis
- Ela é case sensitive (maiúsculas e minúsculas são diferentes.
- char letra = 'a'; → Declaração simples.
- Char trabalha com a tabela ascii.
- Linguagem c usa ponto como separador decimal.
- int nro1 = 10, nro2, nro3;

- Comando printf:
  - Usado para escrever na tela console.
  - mostra tudo que está entre aspas. (" ")
  - (\n) → Usado para pular de linha.
  - (%c) → Caracter → Também pode ser "%d".
  - (%d) → Inteiro
  - (%f) → Flutuante
  - nunca esquecer do ponto e vírgula.
- Comando "scanf":
  - Leitura de dados do teclado e guardar nas variáveis
  - Entre as aspas, definir o tipo de entrada e fora o nome das variáveis.
  - scanf ("tipos de ~~voce~~ entrada", variáveis)
  - Antes do nome das variáveis deve haver um & (É comercial).
  - scanf ("%c", &letra);
  - float e double → (%f).
  - Para ler coloca primeiro um printf e depois um scanf.
- Operador de atribuição:
  - variavel = expressão;

```
int x = 5, y;
y = x;
y = x + 10;
```

- Apenas variáveis podem receber um valor ou expressão.
- `x = y = z = 10` → Todos tem valor 10.
- Caso um char receba um inteiro, será armazenado o valor correspondente na tabela ascii.
- Caso um valor decimal seja colocado em um inteiro, a parte decimal será perdida.

- Constantes:
  - "const" → Palavra usada para declarar uma constante.
  - `const int nro = 10;` → Declaração de uma constante.
  - Uma constante não pode ser mais modificada.
  - "=" → Serial de recebe.
  - `#define` → Usado para definir uma constante.
  - `#define PI 3.1415` → Exemplo.
  - não esquecer de colocar `return 0;` no corpo do programa.
- Operadores aritméticos:
  - Funciona do mesmo modo que na usual.
- Comentários:
  - (//) → Usado para fazer comentários.
  - Usado para explicar o código, contribuindo para o seu possível desenvolvimento futuro.
  - (/* */) → Construção de blocos de comentários.
- Pré e Pós incremento:
  - x++ é igual a x = x+1, adicionar 1 a uma variável.
  - x-- é igual a x = x-1, subtraindo 1.
  - y = x++, é igual a y = x e depois x++.

- y=++x, é igual a x++ e depois y=x.
- Atribuição simplificada:
  - y+=z, é igual a y=y+z.
  - y-=z, y=y-z.
  - y*=z, é igual a y=y*z.
  - x/=z, é igual a x=x/z.
  - x%=z, é igual a x=x%z.

- Anotações:
  - Ao declarar funções e expressões deve haver ponto e vírgula final.