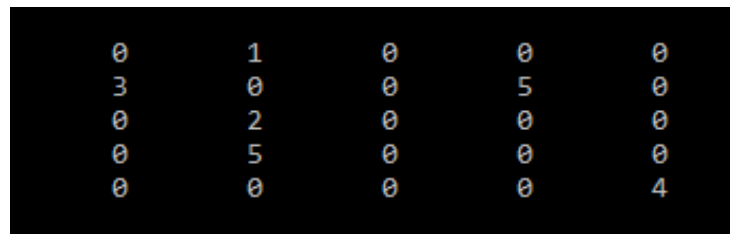


## 3º Exercício: Sudoku Inteligente (Versão 2.0)

### 1- Introdução

Elabore um programa que cria um jogo tipo Sudoku 5x5. Seguindo a regra básica do Sudoku, ou seja, um número não pode se repetir na mesma linha ou coluna, sendo o objetivo do jogador completar todas as posições disponíveis.



|   |   |   |   |   |
|---|---|---|---|---|
| 0 | 1 | 0 | 0 | 0 |
| 3 | 0 | 0 | 5 | 0 |
| 0 | 2 | 0 | 0 | 0 |
| 0 | 5 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 4 |

*Fig 01 Tela inicial do jogo Sudoku Inteligente*

O programa terá início com a apresentação de uma tela 5X5, com 6 posições preenchidas aleatoriamente pelo algoritmo, com valores de 1 até 5 (Cada vez que for dado início ao jogo, oriente seu algoritmo para uma distribuição de peças diferente da anterior tornando aleatório o preenchimento das seis casas). As posições vazias serão preenchidas com o dígito '0' conforme apresentado na Fig 01. Em seguida será solicitado que o jogador preencha um número de dois dígitos representado a posição na matriz na qual ele pretende inserir o primeiro valor. Por exemplo, caso o usuário digite "14", será primeira linha e quarta coluna. Observe que a posição 14 possui o dígito 0, ou seja, é uma jogada permitida. Após inserir a posição será solicitado que insira o valor de 1 até 5 que será armazenado na posição escolhida. O programa deverá, após cada jogada do usuário, sinalizar para o jogador que poderá realizar uma jogada chamada de auxiliar, ou seja, preencher automaticamente uma posição disponível em qualquer posição da matriz (Tabuleiro do Jogo). As jogadas se repetirão até que não exista mais nenhum espaço disponível no tabuleiro. Ao final será apresentada a pontuação do usuário e o mesmo será questionado se deseja jogar novamente.

### 2- Inteligência

Sobre a inteligência do Sudoku, a mesma será implementada por cada jogador na escolha de alguma posição disponível para preenchimento de algum valor possível. Estas posições deverão sempre estar com o valor zero armazenado até o momento que precede a jogada. Caso a posição escolhida pelo algoritmo para realização da jogada auxiliar recaia sobre uma casa cuja inserção de valores de 1 até 5 seja incompatível, esta casa deverá ser marcada com o símbolo 'X'. Observe que a marcação de um símbolo 'X' não sinaliza para o jogador que não existem mais jogadas permitidas no tabuleiro, observe que o teste foi realizado apenas naquela última posição na qual a jogada auxiliar foi realizada.

### 3- Jogadas e Verificações

Os valores válidos para as células do tabuleiro variam de 1 até 5. Qualquer valor negativo passado pelo usuário informa que naquela célula nenhum número válido de 1 até 5 poderá ser colocado segundo as regras do Sudoku. O programa deverá verificar e informar se a jogada é correta. Caso o jogador esteja correto, a célula será preenchida com um 'X', caso contrário será capturado um erro na jogada. Caso o usuário digite uma posição (fora do tabuleiro) o programa também deverá detectar e informar o erro na jogada e consequentemente solicitar nova entrada.

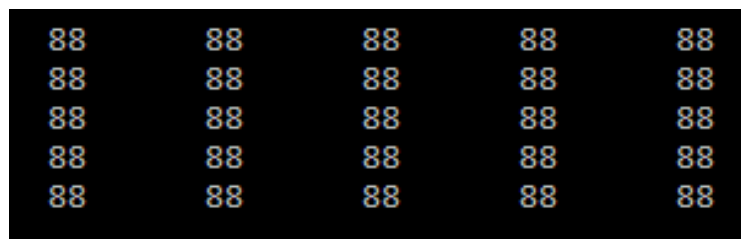
A jogada auxiliar realizada pelo computador será sob comando, ou seja, o programa deverá questionar se o usuário deseja ou não auxílio para a próxima jogada (preenchimento de uma posição vaga).

### 4- Pontuação e Recorde

O programa deverá marcar tanto a pontuação de cada jogada como também o recorde obtido a partir de sucessivas jogadas realizadas pelo mesmo jogador sem sair do programa. A pontuação será feita da seguinte forma, caso o jogador realize todas as jogas sem erros e sem auxílio, receberá 100 pontos (preenchimento completo do tabuleiro sem auxílio). Cada erro subtrai 5 pontos do total e cada auxílio (jogada realizada pelo computador) irá subtrair 2 pontos do total. Cada casa incompleta penaliza com 10 pontos o jogador ao final da partida, ou seja, caso o jogador tenha atingido o limite de erros e restando 5 casas com valor '0', além dos pontos perdidos serão descontados 50 pontos do total.

### 5- Final do Jogo

O jogo finaliza com o preenchimento completo do tabuleiro ou 5 erros consecutivos ou não realizados pelo usuário durante suas jogadas. Será considerada um erro na jogada tanto valores fora do escopo, como números colocados em linhas ou colunas que já tenham o mesmo valor. Caso o jogador consiga preencher todas as posições do tabuleiro o programa deverá apresentar o tabuleiro com todas as posições preenchidas com a pontuação obtida pelo jogador, caso contrário serão todas as posições preenchidas com 'X'. Caso a jogada seja um recorde, o mesmo também deverá ser informado logo abaixo do tabuleiro (A primeira jogada sempre será um recorde). Ao final o programa sempre deverá perguntar se o jogador deseja realizar nova partida.



|    |    |    |    |    |
|----|----|----|----|----|
| 88 | 88 | 88 | 88 | 88 |
| 88 | 88 | 88 | 88 | 88 |
| 88 | 88 | 88 | 88 | 88 |
| 88 | 88 | 88 | 88 | 88 |
| 88 | 88 | 88 | 88 | 88 |

*Fig 02 Tela ao final da partida, jogador obteve 88 pontos.*

Observações:

- Para o preenchimento aleatório da tela inicial utilize a função `rand( )` e `srand( )`. Utilize uma matriz 5x5 de número inteiros para armazenar os dados contidos no tabuleiro.
- Não é permitido usar Strings, apenas o conteúdo constante das aulas de 1-15.
- Use o comando `system("clear")` ou `system("cls");` da biblioteca `<stdlib.h>` que faz a limpeza de toda a tela ao término de cada jogada e da partida.
- Utilizar funções como `telaInicial ( )`, `jogadaAuxiliar( )`, `verificaErro( )` e `verificaFim( )`, além da função `main ( )` para construir o seu programa.
- Comentar sempre que possível os módulos de código desenvolvidos.
- Regras para correção:
  - Uso das boas práticas ensinadas em sala de aula
  - Corretude funcional do algoritmo
  - Capricho na apresentação das telas de interação com o usuário
  - Organização do código

Entrega até dia 11/05/2018

#### Entregar:

- `lab03SEUNOME.cpp`, e

- `lab03SEUNOME.exe`

utilizando o sistema TIDIA no menu Atividades -> Lab03.

#### Cabeçalho:

Obrigatoriamente, **no início** do arquivo fonte, coloque um cabeçalho da seguinte forma:

```
/* Copyright by SEU NOME */
/* Turma 3 */
/* Lab 03: Sudoku Inteligente */
/* Programa compilado com CodeBlocks 17.12 */
```