

# Introdução a Técnicas de Programação - 2014.2

## Descrição de projeto

### FLOOD IT



### Introdução

“Flood it” é um jogo de raciocínio baseado em turnos, cujo objetivo é “inundar” uma área definida por uma matriz 14x14 em no máximo 25 jogadas. A “inundação” ocorre preenchendo as células da matriz com uma única cor. Para isso, o jogador precisa escolher, a cada turno, uma cor a ser “colocada” no canto superior esquerdo. Essa cor irá substituir (preencher), a partir da célula do canto superior esquerdo, toda a área adjacente formada por células que possuem a mesma cor. Se o jogador conseguir preencher toda a matriz em até 25 jogadas, ele ganha, caso contrário, perde.

A dinâmica do jogo e como as células são preenchidas podem ser consultadas em uma versão do jogo online, que se encontra em: <http://floodit.appspot.com/>.

### Descrição do projeto

O projeto do jogo Flood-It deve ser desenvolvido em linguagem C, a ser executado, em sua versão mais simples, através de linha de comando (entrada e saída em um console/terminal). Além das características básicas da mecânica do jogo, o projeto desenvolvido deve implementar os seguintes requisitos funcionais:

- Serão desenvolvidos dois programas diferentes:
  - um programa que gera uma configuração inicial aleatoriamente e grava ela em um arquivo.
  - um programa que lê uma configuração inicial de um arquivo e implementa a lógica do jogo, que é repetir os dois passos seguintes até o término do jogo:
    - leitura de um comando do usuário
    - execução do comando
- Os comandos possíveis são:
  - 'q': interromper o jogo
  - 's': salvar a configuração atual do jogo em arquivo
  - 'o': ler uma configuração de arquivo
  - '0', '1', '2', '3', '4', '5': aplicação de uma das seis cores no canto superior esquerdo (os números de 0 até 5 representam estas cores)
- O término do jogo ocorre em uma das seguintes situações:
  - o jogador conseguiu inundar o tabuleiro em até 25 jogadas, e ele ganhou;
  - o jogador não conseguiu inundar o tabuleiro em até 25 jogadas, e ele perdeu
  - o jogador interrompeu o jogo.

O projeto deve atender também os seguintes critérios de programação:

- Uso de arranjos / matrizes;
- Uso de registros (`struct`);
- Uso de recursão;
- Definição de novos tipos de dados através de `typedef`;
- Modularização do programa em diferentes arquivos (uso de diferentes arquivos `.c` e `.h`, cada um com sua funcionalidade);
- Definição de um padrão de indentação do código fonte e de nomenclatura das sub-rotinas e variáveis, e a adequação a este padrão;
- Documentação adequada do código-fonte.

#### **Observações:**

- O projeto deve ser desenvolvido individualmente ou em duplas (grupos de dois alunos). Não serão permitidos grupos com três ou mais alunos.
- Para facilitar o acompanhamento do projeto, cada dupla deve estar associada a uma única turma de PTP. Ou seja, não serão permitidas duplas formadas por alunos matriculados em turmas de PTP diferentes.
- Cada dupla deve desenvolver sua solução de forma independente das demais. Soluções idênticas serão consideradas plágios e, portanto, sanções serão devidamente aplicadas em todas as duplas com soluções similares.
- Códigos e algoritmos podem ser utilizados da web desde que devidamente referenciados. Caso sejam encontrados trechos de código na web equivalentes aos apresentados pelo grupo sem a devida citação, o código será igualmente

considerado plágio e sanções serão aplicadas ao grupo. Vale salientar que a avaliação será realizada unicamente sobre o código produzido pela dupla. Códigos retirados da web, apesar de permitidos com a devida citação, serão desconsiderados dos critérios de pontuação.

## Acompanhamento e avaliação do projeto

A avaliação do projeto ocorrerá de duas formas:

1. No componente curricular PTP, o projeto será avaliado de forma contínua, ao longo das aulas;
2. No componente curricular ITP, cada aluno será sabatinado individualmente sobre a solução submetida pelo grupo. Ou seja, mesmo que ele tenha feito apenas uma parte do projeto, ele deverá entender como as demais partes foram feitas.

### Avaliação no componente curricular PTP (IMD0012.1)

Esta avaliação compreende a execução e desenvolvimento do projeto em 4 semanas. A cada semana, o grupo deve apresentar uma etapa do projeto desenvolvido, seguindo o calendário abaixo:

#### **ETAPA 1 - semana de 04 de novembro de 2014**

1. Tipos de dados necessários (`typedef`, `structs` e `enums`);
2. Modularização do programa (quais os arquivos `.c` e `.h`);
3. Geração aleatória do tabuleiro (14x14 células);
4. Apresentação da interface inicial do jogo com o tabuleiro gerado.

#### **ETAPA 2 - semana de 11 de novembro de 2014**

5. Leitura da entrada do usuário e atualização do tabuleiro de forma recursiva, preenchendo os adjacentes de mesma cor;

#### **ETAPA 3 - semana de 18 de novembro de 2014**

6. Implementação da condição de término do jogo (máximo 25 rodadas ou tabuleiro completamente preenchido);
7. Salvamento e carregamento do estado de um jogo.

#### **ETAPA 4 - semana de 25 de novembro de 2014**

8. Implementação de elementos extras, definidos pelo próprio grupo.

Exemplos de elementos extras são:

- Estabelecer uma distribuição uniforme na geração das cores no tabuleiro. Ou seja, o tabuleiro contém uma quantidade “quase igual” de células de cada cor (a diferença entre o número de células de cores diferentes não pode ultrapassar um).

- Dar a possibilidade de Salvar e Carregar vários jogos.
- Implementar o jogo usando bibliotecas externas para dar suporte a cores (n-curses, biblioteca de console do windows, etc);
- Implementar o jogo usando uma biblioteca gráfica (GTK, Allegro, SDL, etc);
- Possibilitar a criação de partidas com outros níveis de dificuldades com tabuleiros maiores/menores, número máximos de jogadas e quantidade de cores diferentes;
- Possibilidade de realizar “Undo” (desfazer) um número máximo de vezes (ex: 10 undos), ou mesmo até o início do jogo.

### Sobre as duplas

Os alunos têm ATÉ o dia **04 de novembro de 2014** para comunicar aos professores de ITP e PTP se farão o trabalho em dupla (e a composição da mesma) ou se farão individualmente.

### CrITÉrios de pontuação

O desenvolvimento do projeto aqui descrito vale **100%** da nota da terceira unidade de PTP.

A pontuação da avaliação seguirá os critérios e distribuição abaixo:

- **Atendimento dos requisitos funcionais: 50%**  
a mecânica do jogo está correta? está completa? o jogo salva pontuação máxima? o jogo salva estado do tabuleiro? etc.
- **Uso dos recursos da linguagem C: 20%**  
a dupla demonstrou saber usar de forma adequada os recursos da linguagem C (arranjos, structs, typedefs, recursividade, etc)?
- **Organização do código e documentação: 10%**  
o código está documentado? a indentação e uso de { } seguem um padrão? o programa está devidamente modularizado em diferentes arquivos?
- **Funcionalidades extras: 20%**  
as funcionalidades extras desenvolvidas pela dupla foram suficientemente complexas?

A pontuação a ser dada pelas funcionalidades extras não é definida *a priori*. Cada caso será avaliado em função da complexidade envolvida. Itens extras de baixa complexidade serão desconsiderados na pontuação.

### Avaliação no componente curricular ITP (IMD0012.0)

Esta avaliação ocorrerá após a entrega da versão final o projeto. Ela consiste em uma entrevista individual com os membros do grupo, na qual cada integrante deve demonstrar conhecimento sobre o que foi implementado, como e por quê. O professor de IMD0012.0 irá marcar os horários de entrevista após a entrega dos projetos.

## Critérios de pontuação

O desenvolvimento do projeto aqui descrito vale **ATÉ 2 PONTOS EXTRAS** na menor nota de ITP.

A pontuação da avaliação seguirá os critérios e distribuição abaixo:

- **Domínio da solução apresentada: 50%**
- **Atendimento dos requisitos funcionais: 30%**
- **Funcionalidades extras: 20%**

## Entrega do projeto

O projeto deve ser submetido pelo SIGAA até a data **25 de novembro de 2014** em um **arquivo comprimido (.zip)** contendo os arquivos fontes do projeto (.c e .h) e um arquivo README.TXT. Este arquivo deve ter as seguintes informações:

- O que foi feito (o básico e as funcionalidades extras implementadas);
- O que não foi feito (caso não tenha sido possível implementar alguma funcionalidade);
- O que seria feito diferentemente (quando aprendemos à medida que vamos implementando, por vezes vemos que podíamos ter implementado algo de forma diferente. Assim, esse tópico é para vocês refletirem sobre o que vocês aprenderam durante o processo que, se fossem fazer o mesmo projeto novamente, fariam diferente);
- Como compilar o projeto, deixando explícito se foi utilizada alguma biblioteca externa que precise ser instalada, que versão e quais parâmetros extras são necessários para o compilador.
- Em caso de duplas:
  - Identificação dos autores;
  - Contribuição de cada integrante no desenvolvimento do projeto (quem fez o quê).

## Recomendações diversas:

1. Solução de back-up: não será tolerada a desculpa de que um disco rígido falhou nas avaliações. Assim, é importante manter várias cópias do código-fonte desenvolvido ou usar um sistema de backup automático como o Dropbox, Google Drive, Box ou similares. Uma solução melhor ainda é fazer uso de um sistema de controle de versões como git, e um repositório externo como Github ou Bitbucket.
2. Especificar precisamente a interface e o comportamento de cada sub-rotina. Usar esta informação para guiar o desenvolvimento e documentar o código desenvolvido.