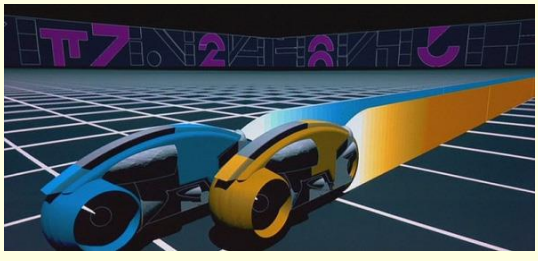
CS 241
Data Organization using C
Project 3: Tron Fall 2019

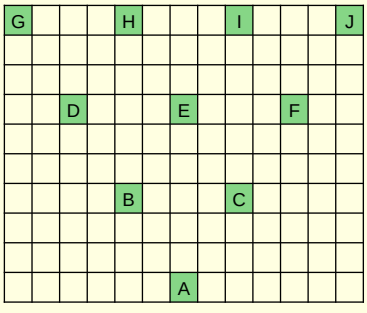
Instructor: **Joel Castellanos**
e-mail: joel@unm.edu
Web: http://cs.unm.edu/~joel/

11/21/2019

Tron *Fall 2019:* Project Outline (1 of 4)

1) The game takes place on an $n \times m$ grid with the minimum and maximum defined in tron.h. The Tourney take place on the maximum grid size.
2) In each *game*, there are 2 teams. Each team is controlled by one student's code or by the instructor's code.
3) Each team starts with 10 Lightcycles in a triangular "bowling pin" formation:





Tron: Project Outline (2 of 4)

4) The north team's cycles are 'A' through 'J' and the south team's 'K' through 'T'. North being at the top of the display.
5) Each turn, each cycle must move in one of the cardinal directions.
6) Every place where a cycle moves it leaves a trail.
7) If a cycle moves into a space with a trail it explodes.
8) If a cycle does not move it explodes.
9) If a cycle moves into the grid perimeter, it explodes.
10) If a cycle moves into the same space as another cycle, the team with the greater total CPU time explodes (in the case of the times being the same, which of the cycles entering the same space explodes is random).

3

3

Tron Project Outline (3 of 4)

11) Once during the game, each cycle may jump one cell.
12) When a cycle explodes, it and its trail is instantly removed from the game.
13) A cycle's trail is the set of spaces from its most recent move back its starting point. Note: due to the jump mechanic this path may be discontinuous at one location.
14) Each team's formation starts centered on either the north or south edge of the board with the row of four cycles being three cells from edge and the row of one cycle nearest the center.
15) No AI may ever call `srand()`. The MCP calls `srand()` with an optional user argument.

4

4

Tron: Project Outline (4 of 4)

16) Each turn, the MCP tells each player the current location of all cycles.
17) Each turn, each team tells the MCP where it moves each of its cycles.
18) If a team fails to give its move instructions before the timeout, then all cycles of that team explode.
19) Timeout for the first turn is 5 seconds.
20) Timeout for each other turn is 0.1 seconds average (not counting turn 1).
21) Walls in the center of the court are given to the player by the `_init()` function.

5

5

MCP / Player Interface

1) Each player has 3 "public" `_init()`, `_move()`,
2) At the start of a new game, the MasterControlProgram (MCP) calls each team's `_init()` function giving it the game configuration:
   a) Each team's name and color.
   b) Each team's initial cycle coordinates.
3) Each timestep, the MCP will call team's `_move()`
   - *Telling* each team the current location of all cycles.
   - *Telling* each team the total CPU time of each team.
   - *Asking* each player where it wants to move each of its cycles.

6

6

Fall 2019: Game Rules

1) Only the MCP may call a player's or `_getName()` functions.
2) Any player that causes the program to exit with a runtime error is disqualified.
3) Any invalid moves cause the cycle to explode.
4) Player code that can trick the MCP without causing the program to crash might be able to give itself an advantage.

7

7

Grading Rubric (75 points total)

[+15 Points]: Your AI wins 3 out of 3 games vs RandomBot.

[+5 Points]: Your AI wins 3 out of 10 games vs LineBot.

[+5 Points]: Your AI wins 5 out of 10 games vs LineBot.

[+5 Points]: Your AI wins 6 out of 10 games vs LineBot.

[+5 Points]: Your AI wins 7 out of 10 games vs LineBot.

[+5 Points]: Your AI wins 8 out of 10 games vs LineBot.

[+5 Points]: Your AI wins 9 out of 10 games vs LineBot.

[+15 Points]: Your AI wins 5 out of 10 games vs LookBot.

[+15 Points]: Your AI wins 9 out of 10 games vs LookBot.

8

8

Tron *Spring 2017:* Extra Credit

Tourney
Thursday
December 12
3:00 - 5:00
CEC Basement Lab
Snacks will be served

+10 Points: Tournament Finalist (including Special Shapes Award).

+25 Points: Bronze tournament Winner.

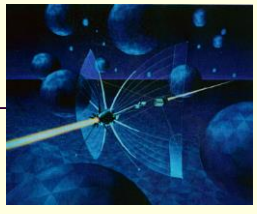
+50 Points: Silver tournament Winner.

+75 Points: Gold tournament Winner.

9

9

Interface between MCP and AI



Each student must implement the three, static (*"public"*) functions:

```
char* firstname_lastname_getName()

void firstname_lastname_init(struct InitData *data)

void firstname_lastname_move(struct MoveData *data)
```

Important: Do not save the addresses:

```
struct InitData *data
struct MoveData *data
```

The memory will be freed after your function returns.
Any *values* you want to save must be copied ***value-by-value*** to `static` local or `static` file-scope variables or structures.

10

10

Cycle Move Order

After all teams have returned from `_move(struct MoveData *data)`, the MCP collects and executes the moves. The AI with the lowest total CPU time is moved first. Within a team, cycles are moved in order of the linked list of cycles in the MoveData structure.

Usually, an AI will not care which of their cycles is moved first, but in rare cases it may matter. For example, one of your cycles destroys itself to clear its path for another of your cycles to enter.

In such cases, reorder the given linked list so that the cycle you want to be moved before another is before that other in the linked list.

tron.h: _getName

`char*` *firstname_lastname*`_getName()`

Returns a pointer to a character array containing your player name.

The returned name may contain any non-whitespace, printable ASCII character (base-10 codes 33 through 126).

If more than one student in the class returns a particular name, the MCP will force each to be unique by appending numbers.

A player's `_getName()` function must always return the same name.

When `_getName()` returns, the MCP will copy the name into its own storage and never again use the returned pointer.

No more than `MAX_PLAYER_NAME_LEN` characters (not including the terminating `'\0'`) will be copied from the returned pointer.

Name must be NULL-terminate.

Compiling Tron

1) Place your AI code in the file: **tron_***yourFirstName_yourLastName*.c
2) In a clean directory with your source code, copy from the class website the files **tron.h**, **libmcp.a** (the MCP library), and **classFunctions.c**.
3) Edit **classFunctions.c** by commenting out each **_getName()**, **_init()** and **_move()** method with you name.
4) Compile all **.c** files in the directory and link with the static MCP library, **libmcp.a**:

```
gcc  *.c -L. -lmcp -lm -lpthread `sdl-config --libs` -lSDL_ttf -o tron
```

This compiles all **.c** files in the directory, links with **libmcp.a** and creates a stand-alone executable with the name **tron**.

13

Running the MCP with your AI

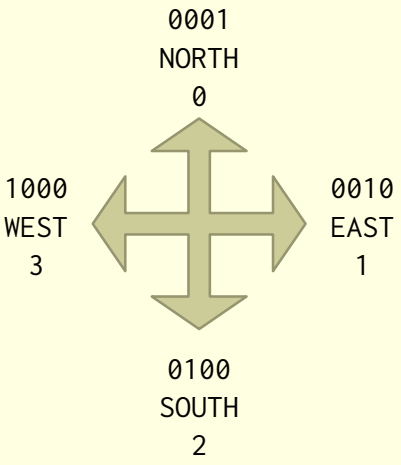
- After you have compiled your code and linked with the MCP, run by entering the command:

```
./tron
```

- With no arguments, **tron** will display a usage screen telling you what arguments to use to specify the, an optional random number seed, which players are to compete, etc.
- Running **tron** with a graphics display will show a realtime SDL (Simple DirectMedia Layer) window.

14

Representing Direction



Given a variable, dirBits, with some bits 1 through 4 turned on if and only if it is possible to move in that direction.

How can you remove a bit?

You can subtract, ***if you first make sure the bit is set***.

You can safely flip the bits of the direction you want to remove, then bitwise AND:

```
dirBits = dirBits & (~bit)
```

15

Representing Direction

```
#define NORTH 1
#define EAST 2
#define SOUTH 4
#define WEST 8
```

given in **tron.h**

By using powers of 2, one integer variable can represent more than one direction. For example:

```
static int getDirBits(int x, int y)
{
  int dirBits = 15;  //All 4 flags on: =N|E|S|W
  if (grid[x][y-1] != WHITE) dirBits -= NORTH;
  if (grid[x][y+1] != WHITE) dirBits -= SOUTH;
  if (grid[x-1][y] != WHITE) dirBits -= EAST;
  if (grid[x+1][y] != WHITE) dirBits -= WEST;
  return dirBits;
}
```

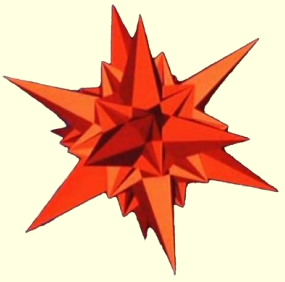
Why not use if, else if, else?

16

Use Many Small Helper Functions

It is often useful to know the number of choices into which a particular cell may.

```
static int getNumberOfOpenDirections(int dirBits)
{
  int openCount = 0;
  if (dirBits & UP) openCount++;
  if (dirBits & DOWN) openCount++;
  if (dirBits & LEFT) openCount++;
  if (dirBits & RIGHT) openCount++;
  return openCount;
}
```



17

17

Unit Test your Helper Functions

```
static void test_getNumberOfOpenDirections()
{
  // 1) Clear the grid.
  // 2) Set, using a short series of assignment statements,
  //    a **few** specific walls and player colored cells
  //    showing in a small area  of the grid.
  // 3) Call getNumberOfOpenDirections() on the
  //     various cases and print results.
  // 4) Verify your results with hand drawing on graph paper.
}
```

18

18

Test Small !

- When developing your AI, change the grid size to something you can print out and actually read.
- After you have the bugs worked out on a small size, THEN try the full size.
- When you find a bug on a full size grid, go back to a small size and try to reproduce it.

19

C keyword: `static`

- In C, a variable declared as **`static`** in a function is initialized once, and retains its value between function calls.
- The default initial value of an uninitialized **`static`** variable is zero.
- If a function or variable is declared **`static`**, it can ***only be accessed in that file***.

Use `static` to protect your variables and function calls from other teams.

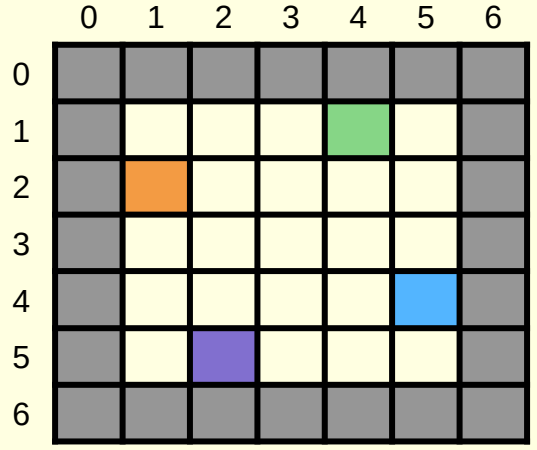
20

Suggested Structure for Grid

Create a *file scope* grid that includes the boundary wall.

Your file scope variables will not be visible to the Master Control Program nor to other student's code.

Shown is a 7x7 grid which includes the boundary wall cells shown in gray.



21

21

Suggested Data Structure for Cycle Trails

```
static int grid[MAX_GRID_WIDTH][MAX_GRID_HEIGHT];
```

File scope

Sized for the biggest possible grid.

- Cycle path must persist between calls to _move.
- The locations of the boundary
- Empty cells must be identifiable.
- Cells containing lightcycle trails must be identifiable.
- When a cycle is terminated, there must be a way of removing its path without disturbing other lightcycle trails.

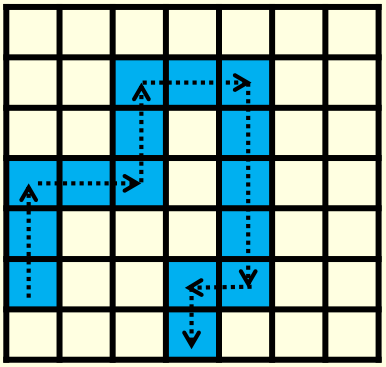
How will you *represent*, *access* and *update* this information?`

22

22

Strategies

- Deep look-ahead (dead ends, open path to far wall, …).
- Attack (nearest, or only within direct sight, or within 10 cells).
- Leave gaps to escape
- Team Formations.
- Trap Building.
- Maximizing open space.
- Efficient Fill.
- Maintaining a goal direction or location through a perturbation.



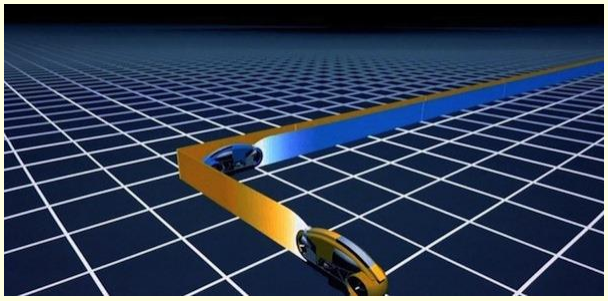
23

Viable Choices and Atari

In the Tron game, *each cycle as at most 7 choices* (counting jump move) of where to move.

It is often useful to know the number of viable choices.

You may, for example, have an expensive look-ahead algorithm. However, if a cycle is in *atari (only one path to survival)*, there is no need to run expensive AI code.



24

Helper: getCycleIndex

- ***What*** might “cycleIdx" be used for?
- ***How*** am I defining it?
- ***Why*** am I defining it in this way?
- Is it needed?
- Are there other ways to define it?

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include "tron.h"

static int getCycleIndex(char cycleName)
{ int cycleIdx = cycleName - 'A';
  if ((cycleIdx < 0) || (cycleIdx >= CYCLE_COUNT))
  {
    printf("JoelRobot.c: ***ERROR*** getCycleIndex(%d)\n",
            cycleIdx);
    exit(0);
  }
  return cycleIdx;
}
```

Gotta Love That `printf`

Prints up to a 20×20 grid within a single, default size PuTTY window.
Try printing your grid just before and just after a cycle crash.

```
static void printGrid(void)
{ int x, y;
  for (y=0; y<gridHeight; y++)
  { for (x=0; x<gridWidth; x++)
    {
      printf("%3d ", grid[x][y]);
    }
    printf("\n");
  }
}
```

Gotta Love That `printf`

Cell encodes enter direction

```
256 256 256 256 256 256 256 256 256 256 256 256 256 256 256
256   0   0   0  64  64   1   0   2   0   0   0   0   0 256
256   0   0   0   0  64   1   2   2   0   0   0   0   0 256
256   0   0   0   0  64   1   3   0   0   0   0   0   0 256
256   0   0   0   0  64  64   3   0   0   0   0   0   0 256
256   0   0   0   0   0  64   3   0   0   0   0   0   0 256
256  64  64  64  64  64  64   3   8   8   8   8   8   8 256
256  32  32  32  32  32  32   3   8   0   0   0   0   0 256
256  32   0   0   0   0  32   3   8   0   0   0   0   0 256
256   0   0   0   0   0  32   3   8   0   0   0   0   0 256
256   0   0   0   0   0  32   3   8   0   0   0   0   0 256
256   0   0   0   0   0  32   3   0   0   0   0   0   0 256
256   0   0   0   0   0  32   3   0   0   0   0   0   0 256
256   0   0   0   0   0  32   0   0   0   0   0   0   0 256
256 256 256 256 256 256 256 256 256 256 256 256 256 256 256
```


27