Practical 10
Space Invaders (Part 13)

Step 1

Now, the program has to determine which invader should fire and when. To do so, you have to follow the following steps:

For each iteration of the main loop:

- Generate a random number between 0 and 2,047.
- If the random number is greater than or equal to the total number of invaders (`INVADER_COUNT`), no shot should be connected to an invader.
- If the random number is lower than the total number of invaders, this number should be used as an index in order to determine the address of an invader (e.g. the 0-indexed invader is located at the address `Invaders`, the 1-indexed invader is located at the address `Invaders+SIZE_OF_SPRITE`, etc.), and if this invader is displayed, connect it to an available shot (by calling **`ConnectInvaderShot`**).

Use the following subroutine in order to generate the random number (do not try to understand it):

```
Random              move.l  \old,d0
                    muls.w  #16807,d0
                    and.l   #$7fffffff,d0
                    move.l  d0,\old
                    lsr.l   #4,d0
                    and.l   #$7ff,d0
                    rts
\old                dc.l    425625
```

Write down the **`NewInvaderShot`** subroutine that follows the above steps and call it in the main loop of your source code.

```
Main                jsr     InitInvaders
                    jsr     InitInvaderShots

\loop               jsr     PrintShip
                    jsr     PrintShipShot
                    jsr     PrintInvaders
                    jsr     BufferToScreen

                    jsr     DestroyInvaders

                    jsr     MoveShip
                    jsr     MoveInvaders
                    jsr     MoveShipShot

                    jsr     NewShipShot
                    jsr     NewInvaderShot

                    jsr     SpeedInvaderUp

                    bra     \loop
```

Step 2

Using the **PrintInvaders** subroutine as a model, write the **PrintInvaderShots** subroutine that displays all the invader shots.

Then, call this subroutine in your main loop:

```
Main                jsr     InitInvaders
                    jsr     InitInvaderShots

\loop               jsr     PrintShip
                    jsr     PrintShipShot
                    jsr     PrintInvaders
                    jsr     PrintInvaderShots
                    jsr     BufferToScreen

                    jsr     DestroyInvaders

                    jsr     MoveShip
                    jsr     MoveInvaders
                    jsr     MoveShipShot

                    jsr     NewShipShot
                    jsr     NewInvaderShot

                    jsr     SpeedInvaderUp

                    bra     \loop
```

Run your main program and check that five shots are displayed. For the time being, they are frozen. We will set them in motion in the next step.

Step 3

In this step, we are going to set the invader shots in motion. To begin with, let us define a constant that holds the step increment of a shot (as we did for the ship and the invaders).

```
                    ; Step increment in pixels
                    ; ----------------------------

SHIP_STEP           equ     4                           ; Ship step increment
SHIP_SHOT_STEP      equ     4                           ; Ship shot step increment
INVADER_SHOT_STEP   equ     1                           ; Invader shot step increment
; ...
```

Use the **MoveShipShot** subroutine (which handles the displacement of the ship shot) as a model to help you write the **MoveInvaderShots** subroutine that moves the invader shots. Obviously, when a shot has reached the bottom of the screen, it becomes invisible and so available. Be careful, **MoveShipShot** handles one sprite only (the ship shot), but **MoveInvaderShots** handles several sprites (all the invader shots located from the `InvaderShots` address).

Tips:

- In order to move a sprite, you can call the **`MoveSprite`** subroutine that was written in a previous step.
- So far, the animation of the sprites (swapping bitmaps 1 and 2) has not been required. We will do this in the next step. So, only the bitmap 1 will be displayed.

Finally, call this subroutine in your main loop and test it.

```
Main                jsr     InitInvaders
                    jsr     InitInvaderShots

\loop               jsr     PrintShip
                    jsr     PrintShipShot
                    jsr     PrintInvaders
                    jsr     PrintInvaderShots
                    jsr     BufferToScreen

                    jsr     DestroyInvaders

                    jsr     MoveShip
                    jsr     MoveInvaders
                    jsr     MoveShipShot
                    jsr     MoveInvaderShots

                    jsr     NewShipShot
                    jsr     NewInvaderShot

                    jsr     SpeedInvaderUp

                    bra     \loop
```