```
//************************************
// Name: paravia.c
// Description:This is a port of the original TRS-80 BASIC code for Santa Paravi
a and Fiumaccio, (C) 1979 George Blank (used with permission).
// By: Thomas Knox
//
// Inputs:N/A
//
// Returns:N/A
//
// Assumes:Should compile and run on any system with an ANSI-C compiler.
//
// Side Effects:N/A
//
//This code is copyrighted and has
// limited warranties.Please see http://www.Planet-Source-Code.com/vb/scripts/Sh
owCode.asp?txtCodeId=7183&lngWId=3
//for details.
//************************************

/*****************************************************************************
 ** **
 ** Santa Paravia & Fiumaccio. Translated from the original TRS-80 BASIC **
 ** source code into C by Thomas Knox <tknox@mac.com>.**
 ** **
 ** Original program (C) 1979 by George Blank**
 ** <gwblank@postoffice.worldnet.att.net>**
 ** **
 *****************************************************************************/
/*
Copyright (C) 2000 Thomas Knox
Portions Copyright (C) 1979 by George Blank, used with permission.
This program is free software; you can redistribute it and/or
modify it under the terms of the GNU General Public License
as published by the Free Software Foundation; either version 2
of the License, or (at your option) any later version.
This program is distributed in the hope that it will be useful,
but WITHOUT ANY WARRANTY; without even the implied warranty of
MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the
GNU General Public License for more details.
You should have received a copy of the GNU General Public License
along with this program; if not, write to the Free Software
Foundation, Inc., 59 Temple Place - Suite 330, Boston, MA 02111-1307, USA.
Thomas Knox
tknox@mac.com
*/
/* Declare our standard C headers. */
#include <stdio.h>
#include <string.h>
#include <time.h>
#include <stdlib.h>
#include <curses.h>
/* Declare an enum to emulate a Boolean. */
enum TrueFalse {True, False};
typedef enum TrueFalse boolean;
/* Declare our player definition. */
struct Player
{
int Cathedral, Clergy, CustomsDuty, CustomsDutyRevenue, DeadSerfs;
int Difficulty, FleeingSerfs, GrainDemand, GrainPrice, GrainReserve;
int Harvest, IncomeTax, IncomeTaxRevenue, RatsAte;
int Justice, JusticeRevenue, Land, Marketplaces, MarketRevenue;
```



```
int Merchants, MillRevenue, Mills, NewSerfs, Nobles, OldTitle, Palace;
int Rats, SalesTax, SalesTaxRevenue, Serfs, SoldierPay, Soldiers, TitleNum;
int TransplantedSerfs, Treasury, WhichPlayer, Year, YearOfDeath;
char City[15], Name[25], Title[15];
float PublicWorks, LandPrice;
boolean InvadeMe, IsBankrupt, IsDead, IWon, MaleOrFemale, NewTitle;
} Player;
typedef struct Player player;
/* Declare our list of cities. */
char CityList[7][15] = {"Santa Paravia", "Fiumaccio", "Torricella", "Molinetto",
"Fontanile", "Romanga", "Monterana"};
/* Declare our male titles. */
char MaleTitles[8][15] = {"Sir", "Baron", "Count", "Marquis", "Duke",
 "Grand Duke", "Prince", "* H.R.H. King"};
/* Declare our female titles. */
char FemaleTitles[8][15] = {"Lady", "Baroness", "Countess", "Marquise",
"Duchess", "Grand Duchess", "Princess",
            "* H.R.H. Queen"};
/* Our prototypes. */
int main(void);
int Random(int);
void InitializePlayer(player*, int, int, int, char*, boolean);
void AddRevenue(player *);
int AttackNeighbor(player *, player *);
void BuyCathedral(player *);
void BuyGrain(player *);
void BuyLand(player *);
void BuyMarket(player *);
void BuyMill(player *);
void BuyPalace(player *);
void BuySoldiers(player *);
int limit10(int, int);
boolean CheckNewTitle(player *);
void GenerateHarvest(player *);
void GenerateIncome(player *);
void ChangeTitle(player *);
void NewLandAndGrainPrices(player *);
void PrintGrain(player *);
int ReleaseGrain(player *);
void SeizeAssets(player *);
void SellGrain(player *);
void SellLand(player *);
void SerfsDecomposing(player *, float);
void SerfsProcreating(player *, float);
void PrintInstructions(void);
void PlayGame(player [], int);
void NewTurn(player *, int, player [], player *);
void BuySellGrain(player *);
void AdjustTax(player *);
void DrawMap(player *);
void StatePurchases(player *, int, player []);
void ShowStats(player [], int);
void ImDead(player *);
int main(void)
{
player MyPlayers[6];
int NumOfPlayers, i, level;
char string[255], name[25];
boolean MorF;
/* Initialize the random number generator seed. */
srand(time(NULL));
/* Start the game. */
```

```
printf("Santa Paravia and Fiumaccio\n");
printf("\nDo you wish instructions (Y or N)? ");
fgets(string, 254, stdin);
if(string[0] == 'y' || string[0] == 'Y')
    PrintInstructions();
printf("How many people want to play (1 to 6)? ");
fgets(string, 254, stdin);
NumOfPlayers = (int)atoi(string);
if(NumOfPlayers < 1 || NumOfPlayers > 6)
{
printf("Thanks for playing.\n");
return(0);
}
printf("What will be the difficulty of this game:\n1. Apprentice\n");
printf("2. Journeyman\n3. Master\n4. Grand Master\n\nChoose: ");
fgets(string, 254, stdin);
level = (int)atoi(string);
if(level < 1)
level = 1;
if(level > 4)
level = 4;
for(i = 0; i < NumOfPlayers; i++)
{
printf("Who is the ruler of %s? ", CityList[i]);
fgets(name, 24, stdin);
    /* Strip off the trailing \n. */
    name[strlen(name) - 1] = '\0';
printf("Is %s a man or a woman (M or F)? ", name);
fgets(string, 3, stdin);
if(string[0] == 'm' || string[0] == 'M')
MorF = True;
else
MorF = False;
InitializePlayer(&MyPlayers[i], 1400, i, level, name, MorF);
}
/* Enter the main game loop. */
PlayGame(MyPlayers, NumOfPlayers);
/* We're finished. */
return(0);
}
/*****************************************************************************
 ** This function will take a parameter Hi and return a random integer**
 ** between 0 and Hi.**
 *****************************************************************************/
int Random(int Hi)
{
float RanNum;
RanNum = (float)rand();
RanNum /= (float)RAND_MAX;
RanNum *= (float)Hi;
return((int)RanNum);
}
void InitializePlayer(player *Me, int year, int city, int level, char *name,
     boolean MorF)
{
/* This is pretty straightforward. */
Me->Cathedral = 0;
strcpy(Me->City, CityList[city]);
Me->Clergy = 5;
Me->CustomsDuty = 25;
Me->Difficulty = level;
Me->GrainPrice = 25;
```



```
Me->GrainReserve = 5000;
Me->IncomeTax = 5;
Me->IsBankrupt = False;
Me->IsDead = False;
Me->IWon = False;
Me->Justice = 2;
Me->Land = 10000;
Me->LandPrice = 10.0;
Me->MaleOrFemale = MorF;
Me->Marketplaces = 0;
Me->Merchants = 25;
Me->Mills = 0;
strcpy(Me->Name, name);
Me->Nobles = 4;
Me->OldTitle = 1;
Me->Palace = 0;
Me->PublicWorks = 1.0;
Me->SalesTax = 10;
Me->Serfs = 2000;
Me->Soldiers = 25;
Me->TitleNum = 1;
if(Me->MaleOrFemale == True)
strcpy(Me->Title, MaleTitles[0]);
else
strcpy(Me->Title, FemaleTitles[0]);
if(city == 6) strcpy(Me->Title, "Baron");
Me->Treasury = 1000;
Me->WhichPlayer = city;
Me->Year = year;
Me->YearOfDeath = year + 20 + Random(35);
return;
}
void AddRevenue(player *Me)
{
Me->Treasury += (Me->JusticeRevenue + Me->CustomsDutyRevenue);
Me->Treasury += (Me->IncomeTaxRevenue + Me->SalesTaxRevenue);
/* Penalize deficit spending. */
if(Me->Treasury < 0)
Me->Treasury = (int)((float)Me->Treasury * 1.5);
/* Will a title make the creditors happy (for now)? */
if(Me->Treasury < (-10000 * Me->TitleNum))
Me->IsBankrupt = True;
return;
}
int AttackNeighbor(player *Me, player *Him)
{
int LandTaken;
int deadsoldiers = 0;
if(Me->WhichPlayer == 7)
    LandTaken = Random(9000) + 1000;
else
    LandTaken = (Me->Soldiers * 1000) - (Me->Land / 3);
if(LandTaken > (Him->Land - 5000))
    LandTaken = (Him->Land - 5000) / 2;
Me->Land += LandTaken;
Him->Land -= LandTaken;
printf("\a\n%s %s of %s invades and seizes %d hectares of land!\n",
        Me->Title, Me->Name, Me->City, LandTaken);
deadsoldiers = Random(40);
if(deadsoldiers > (Him->Soldiers - 15))
    deadsoldiers = Him->Soldiers - 15;
Him->Soldiers -= deadsoldiers;
```

```
printf("%s %s loses %d soldiers in battle.\n", Him->Title, Him->Name,
        deadsoldiers);
return(LandTaken);
}
void BuyCathedral(player *Me)
{
Me->Cathedral += 1;
Me->Clergy += Random(6);
Me->Treasury -= 5000;
Me->PublicWorks += 1.0;
return;
}
void BuyGrain(player *Me)
{
char string[256];
int HowMuch;
printf("How much grain do you want to buy (0 to specify a total)? ");
fgets(string, 255, stdin);
HowMuch = (int)atoi(string);
if(HowMuch == 0)
{
    printf("How much total grain do you wish? ");
    fgets(string, 255, stdin);
    HowMuch = (int)atoi(string);
    HowMuch -= Me->GrainReserve;
    if(HowMuch < 0)
    {
        printf("Invalid total amount.\n\n");
        return;
    }
}
Me->Treasury -= (HowMuch * Me->GrainPrice / 1000);
Me->GrainReserve += HowMuch;
return;
}
void BuyLand(player *Me)
{
char string[256];
int HowMuch;
printf("How much land do you want to buy? ");
fgets(string, 255, stdin);
HowMuch = (int)atoi(string);
Me->Land += HowMuch;
Me->Treasury -= (int)(((float)HowMuch * Me->LandPrice));
return;
}
void BuyMarket(player *Me)
{
Me->Marketplaces += 1;
Me->Merchants += 5;
Me->Treasury -= 1000;
Me->PublicWorks += 1.0;
return;
}
void BuyMill(player *Me)
{
Me->Mills += 1;
Me->Treasury -= 2000;
Me->PublicWorks += 0.25;
return;
}
void BuyPalace(player *Me)
```



```
{
Me->Palace += 1;
Me->Nobles += Random(2);
Me->Treasury -= 3000;
Me->PublicWorks += 0.5;
return;
}
void BuySoldiers(player *Me)
{
Me->Soldiers += 20;
Me->Serfs -= 20;
Me->Treasury -= 500;
}
int limit10(int num, int denom)
{
register int val;
val = num / denom;
return(val > 10 ? 10 : val);
}
boolean CheckNewTitle(player *Me)
{
int Total;
/* Tally up our success so far . . . . */
Total = limit10(Me->Marketplaces, 1);
Total += limit10(Me->Palace, 1);
Total += limit10(Me->Cathedral, 1);
Total += limit10(Me->Mills, 1);
Total += limit10(Me->Treasury, 5000);
Total += limit10(Me->Land, 6000);
Total += limit10(Me->Merchants, 50);
Total += limit10(Me->Nobles, 5);
Total += limit10(Me->Soldiers, 50);
Total += limit10(Me->Clergy, 10);
Total += limit10(Me->Serfs, 2000);
Total += limit10((int)(Me->PublicWorks * 100.0), 500);
Me->TitleNum = (Total / Me->Difficulty) - Me->Justice;
if(Me->TitleNum > 7)
Me->TitleNum = 7;
if(Me->TitleNum < 0)
Me->TitleNum = 0;
/* Did we change (could be backwards or forwards)? */
if(Me->TitleNum > Me->OldTitle)
{
Me->OldTitle = Me->TitleNum;
ChangeTitle(Me);
    printf("\aGood news! %s has achieved the rank of %s\n\n", Me->Name,
            Me->Title);
return(True);
}
Me->TitleNum = Me->OldTitle;
return(False);
}
void GenerateHarvest(player *Me)
{
Me->Harvest = (Random(5) + Random(6)) / 2;
Me->Rats = Random(50);
Me->GrainReserve = ((Me->GrainReserve * 100) -
        (Me->GrainReserve * Me->Rats)) / 100;
return;
}
void GenerateIncome(player *Me)
{
```

```
float y;
int revenues = 0;
char string[256];
Me->JusticeRevenue = (Me->Justice * 300 - 500) * Me->TitleNum;
switch(Me->Justice)
{
    case 1: strcpy(string, "Very Fair"); break;
    case 2: strcpy(string, "Moderate"); break;
    case 3: strcpy(string, "Harsh"); break;
    case 4: strcpy(string, "Outrageous");
}
y = 150.0 - (float)Me->SalesTax - (float)Me->CustomsDuty -
    (float)Me->IncomeTax;
if(y < 1.0)
y = 1.0;
y /= 100.0;
Me->CustomsDutyRevenue = Me->Nobles * 180 + Me->Clergy * 75 +
     Me->Merchants * 20 * y;
Me->CustomsDutyRevenue += (int)(Me->PublicWorks * 100.0);
Me->CustomsDutyRevenue = (int)((float)Me->CustomsDuty / 100.0 *
         (float)Me->CustomsDutyRevenue);
Me->SalesTaxRevenue = Me->Nobles * 50 + Me->Merchants * 25 +
     (int)(Me->PublicWorks * 10.0);
Me->SalesTaxRevenue *= (y * (5 - Me->Justice) * Me->SalesTax);
Me->SalesTaxRevenue /= 200;
Me->IncomeTaxRevenue = Me->Nobles * 250 + (int)(Me->PublicWorks * 20.0);
Me->IncomeTaxRevenue += (10 * Me->Justice * Me->Nobles * y);
Me->IncomeTaxRevenue *= Me->IncomeTax;
Me->IncomeTaxRevenue /= 100;
revenues = Me->CustomsDutyRevenue + Me->SalesTaxRevenue +
    Me->IncomeTaxRevenue + Me->JusticeRevenue;
printf("State revenues %d gold florins.\n", revenues);
printf("Customs Duty\tSales Tax\tIncome Tax\tJustice\n");
printf("%d\t\t%d\t\t%d\t\t%d %s\n", Me->CustomsDutyRevenue,
        Me->SalesTaxRevenue, Me->IncomeTaxRevenue,
        Me->JusticeRevenue, string);
return;
}
void ChangeTitle(player *Me)
{
if(Me->MaleOrFemale == True)
strcpy(Me->Title, MaleTitles[Me->TitleNum]);
else
strcpy(Me->Title, FemaleTitles[Me->TitleNum]);
if(Me->TitleNum == 7)
{
Me->IWon = True;
return;
}
return;
}
void NewLandAndGrainPrices(player *Me)
{
float x, y, MyRandom;
int h;
/* Generate an offset for use in later int->float conversions. */
MyRandom = (float)((float)rand() / (float)RAND_MAX);
/* If you think this C code is ugly, you should see the original BASIC. */
x = (float)Me->Land;
y = (((float)Me->Serfs - (float)Me->Mills) * 100.0) * 5.0;
if(y < 0.0)
y = 0.0;
```



```
if(y < x)
x = y;
y = (float)Me->GrainReserve * 2.0;
if(y < x)
x = y;
y = (float)Me->Harvest + (MyRandom - 0.5);
h = (int)(x * y);
Me->GrainReserve += h;
Me->GrainDemand = (Me->Nobles * 100) + (Me->Cathedral * 40) +
     (Me->Merchants * 30);
Me->GrainDemand += ((Me->Soldiers * 10) + (Me->Serfs * 5));
Me->LandPrice = (3.0 * (float)Me->Harvest + (float)Random(6) + 10.0) / 10.0
if(h < 0)
    h *= -1;
if(h < 1)
y = 2.0;
else
{
y = (float)((float)Me->GrainDemand / (float)h);
if(y > 2.0)
y = 2.0;
}
if(y < 0.8)
y = 0.8;
Me->LandPrice *= y;
if(Me->LandPrice < 1.0) Me->LandPrice = 1.0;
Me->GrainPrice = (int)(((6.0 - (float)Me->Harvest) * 3.0 + (float)Random(5)
           + (float)Random(5)) * 4.0 * y);
Me->RatsAte = h;
return;
}
void PrintGrain(player *Me)
{
switch(Me->Harvest)
{
    case 0:
case 1: printf("Drought. Famine Threatens. "); break;
case 2: printf("Bad Weather. Poor Harvest. "); break;
case 3: printf("Normal Weather. Average Harvest. "); break;
case 4: printf("Good Weather. Fine Harvest. "); break;
case 5: printf("Excellent Weather. Great Harvest! "); break;
}
return;
}
int ReleaseGrain(player *Me)
{
double xp, zp;
float x, z;
char string[256];
int HowMuch, Maximum, Minimum;
boolean IsOK;
IsOK = False;
Minimum = Me->GrainReserve / 5;
Maximum = (Me->GrainReserve - Minimum);
while(IsOK == False)
{
printf("How much grain will you release for consumption?\n");
printf("1 = Minimum (%d), 2 = Maximum(%d), or enter a value: ",
           Minimum, Maximum);
fgets(string, 255, stdin);
HowMuch = (int)atoi(string);
if(HowMuch == 1)
```

```
        HowMuch = Minimum;
    if(HowMuch == 2)
        HowMuch = Maximum;
/* Are we being a Scrooge? */
    if(HowMuch < Minimum)
printf("You must release at least 20%% of your reserves.\n");
/* Whoa. Slow down there son. */
    else if(HowMuch > Maximum)
printf("You must keep at least 20%%.\n");
    else
    IsOK = True;
}
Me->SoldierPay = Me->MarketRevenue = Me->NewSerfs = Me->DeadSerfs = 0;
Me->TransplantedSerfs = Me->FleeingSerfs = 0;
Me->InvadeMe = False;
Me->GrainReserve -= HowMuch;
z = (float)HowMuch / (float)Me->GrainDemand - 1.0;
if(z > 0.0)
z /= 2.0;
if(z > 0.25)
z = z / 10.0 + 0.25;
zp = 50.0 - (double)Me->CustomsDuty - (double)Me->SalesTax -
    (double)Me->IncomeTax;
if(zp < 0.0)
zp *= (double)Me->Justice;
zp /= 10.0;
if(zp > 0.0)
zp += (3.0 - (double)Me->Justice);
z += ((float)zp / 10.0);
if(z > 0.5)
z = 0.5;
if(HowMuch < (Me->GrainDemand - 1))
{
x = ((float)Me->GrainDemand - (float)HowMuch) /
    (float)Me->GrainDemand * 100.0 - 9.0;
xp = (double)x;
if(x > 65.0)
x = 65.0;
if(x < 0.0)
{
xp = 0.0;
x = 0.0;
}
SerfsProcreating(Me, 3.0);
SerfsDecomposing(Me, xp + 8.0);
}
else
{
SerfsProcreating(Me, 7.0);
SerfsDecomposing(Me, 3.0);
if((Me->CustomsDuty + Me->SalesTax) < 35)
Me->Merchants += Random(4);
if(Me->IncomeTax < Random(28))
{
Me->Nobles += Random(2);
Me->Clergy += Random(3);
}
if(HowMuch > (int)((float)Me->GrainDemand * 1.3))
{
zp = (double)Me->Serfs / 1000.0;
z = ((float)HowMuch - (float)(Me->GrainDemand)) /
(float)Me->GrainDemand * 10.0;
```



```
z *= ((float)zp * (float)Random(25));
z += (float)Random(40);
    Me->TransplantedSerfs = (int)z;
    Me->Serfs += Me->TransplantedSerfs;
    printf("%d serfs move to the city\n", Me->TransplantedSerfs);
zp = (double)z;
z = ((float)zp * (float)rand()) / (float)RAND_MAX;
if(z > 50.0)
z = 50.0;
Me->Merchants += (int)z;
Me->Nobles++;
Me->Clergy += 2;
}
}
if(Me->Justice > 2)
{
Me->JusticeRevenue = Me->Serfs / 100 * (Me->Justice - 2) *
     (Me->Justice - 2);
Me->JusticeRevenue = Random(Me->JusticeRevenue);
Me->Serfs -= Me->JusticeRevenue;
Me->FleeingSerfs = Me->JusticeRevenue;
    printf("%d serfs flee harsh justice\n", Me->FleeingSerfs);
}
Me->MarketRevenue = Me->Marketplaces * 75;
if(Me->MarketRevenue > 0)
{
Me->Treasury += Me->MarketRevenue;
    printf("Your market earned %d florins.\n", Me->MarketRevenue);
}
Me->MillRevenue = Me->Mills * (55 + Random(250));
if(Me->MillRevenue > 0)
{
Me->Treasury += Me->MillRevenue;
    printf("Your woolen mill earned %d florins.\n", Me->MillRevenue);
}
Me->SoldierPay = Me->Soldiers * 3;
Me->Treasury -= Me->SoldierPay;
printf("You paid your soldiers %d florins.\n", Me->SoldierPay);
printf("You have %d serfs in your city.\n", Me->Serfs);
printf("(Press ENTER): ");
fgets(string, 255, stdin);
if((Me->Land / 1000) > Me->Soldiers)
{
Me->InvadeMe = True;
return(3);
}
if((Me->Land / 500) > Me->Soldiers)
{
Me->InvadeMe = True;
return(3);
}
return(0);
}
void SeizeAssets(player *Me)
{
char string[256];
Me->Marketplaces = 0;
Me->Palace = 0;
Me->Cathedral = 0;
Me->Mills = 0;
Me->Land = 6000;
Me->PublicWorks = 1.0;
```

```
Me->Treasury = 100;
Me->IsBankrupt = False;
printf("\n\n%s %s is bankrupt.\n", Me->Title, Me->Name);
printf("\nCreditors have seized much of your assets.\n");
printf("\n(Press ENTER): ");
fgets(string, 255, stdin);
return;
}
void SellGrain(player *Me)
{
char string[256];
int HowMuch;
printf("How much grain do you want to sell? ");
fgets(string, 255, stdin);
HowMuch = (int)atoi(string);
if(HowMuch > Me->GrainReserve)
{
    printf("You don't have it.\n");
return;
}
Me->Treasury += (HowMuch * Me->GrainPrice / 1000);
Me->GrainReserve -= HowMuch;
return;
}
void SellLand(player *Me)
{
char string[256];
int HowMuch;
printf("How much land do you want to sell? ");
fgets(string, 255, stdin);
HowMuch = (int)atoi(string);
if(HowMuch > (Me->Land - 5000))
{
    printf("You can't sell that much\n");
return;
}
Me->Land -= HowMuch;
Me->Treasury += (int)(((float)HowMuch * Me->LandPrice));
return;
}
void SerfsDecomposing(player *Me, float MyScale)
{
int absc;
float ord;
absc = (int)MyScale;
ord = MyScale - (float)absc;
Me->DeadSerfs = (int)((((float)Random(absc) + ord) * (float)Me->Serfs) /
        100.0);
Me->Serfs -= Me->DeadSerfs;
printf("%d serfs die this year.\n", Me->DeadSerfs);
return;
}
void SerfsProcreating(player *Me, float MyScale)
{
int absc;
float ord;
absc = (int)MyScale;
ord = MyScale - (float)absc;
Me->NewSerfs = (int)((((float)Random(absc) + ord) * (float)Me->Serfs) /
        100.0);
Me->Serfs += Me->NewSerfs;
printf("%d serfs born this year.\n", Me->NewSerfs);
```



```
return;
}
void PrintInstructions(void)
{
    char string[256];

    printf("Santa Paravia and Fiumaccio\n\n");
    printf("You are the ruler of a 15th century Italian city state.\n");
    printf("If you rule well, you will receive higher titles. The\n");
    printf("first player to become king or queen wins. Life expectancy\n");
    printf("then was brief, so you may not live long enough to win.\n");
    printf("The computer will draw a map of your state. The size\n");
    printf("of the area in the wall grows as you buy more land. The\n");
    printf("size of the guard tower in the upper left corner shows\n");
    printf("the adequacy of your defenses. If it shrinks, equip more\n");
    printf("soldiers! If the horse and plowman is touching the top of the wall,\n");
    printf("all your land is in production. Otherwise you need more\n");
    printf("serfs, who will migrate to your state if you distribute\n");
    printf("more grain than the minimum demand. If you distribute less\n");
    printf("grain, some of your people will starve, and you will have\n");
    printf("a high death rate. High taxes raise money, but slow down\n");
    printf("economic growth. (Press ENTER to begin game)\n");
    fgets(string, 255, stdin);
    return;
}
void PlayGame(player MyPlayers[6], int NumOfPlayers)
{
    boolean AllDead, Winner;
    int i, WinningPlayer = 0;
    player Baron;
    AllDead = False;
    Winner = False;
    InitializePlayer(&Baron, 1400, 6, 4, "Peppone", True);
    while(AllDead == False && Winner == False)
    {
        for(i = 0; i < NumOfPlayers; i++)
            if(MyPlayers[i].IsDead == False)
                NewTurn(&MyPlayers[i], NumOfPlayers, MyPlayers,
                        &Baron);
        AllDead = True;
        for(i = 0; i < NumOfPlayers; i++)
            if(AllDead == True && MyPlayers[i].IsDead == False)
                AllDead = False;
        for(i = 0; i < NumOfPlayers; i++)
            if(MyPlayers[i].IWon == True)
            {
                Winner = True;
                WinningPlayer = i;
            }
    }
    if(AllDead == True)
    {
        printf("The game has ended.\n");
        return;
    }
    printf("Game Over. %s %s wins.\n", MyPlayers[WinningPlayer].Title,
MyPlayers[WinningPlayer].Name);
    return;
}
void NewTurn(player *Me, int HowMany, player MyPlayers[6], player *Baron)
{
    int i;
```

Printed by Fitzgerald Bro


```
    GenerateHarvest(Me);
    NewLandAndGrainPrices(Me);
    BuySellGrain(Me);
    ReleaseGrain(Me);
    if(Me->InvadeMe == True)
    {
        for(i = 0; i < HowMany; i++)
            if(i != Me->WhichPlayer)
                if(MyPlayers[i].Soldiers > (Me->Soldiers * 2.4))
                {
                    AttackNeighbor(&MyPlayers[i], Me);
                    i = 9;
                }
        if(i != 9)
            AttackNeighbor(Baron, Me);
    }
    AdjustTax(Me);
    DrawMap(Me);
    StatePurchases(Me, HowMany, MyPlayers);
    CheckNewTitle(Me);

    Me->Year++;
    if(Me->Year == Me->YearOfDeath)
        ImDead(Me);
    if(Me->TitleNum >= 7)
        Me->IWon = True;
}
void BuySellGrain(player *Me)
{
    boolean Finished;
    char string[256];
    Finished = False;
    while(Finished == False)
    {
        printf("\nYear %d\n", Me->Year);
        printf("\n%s %s\n\n", Me->Title, Me->Name);
        printf("Rats ate %d%% of your grain reserves.\n", Me->Rats);
        PrintGrain(Me);
        printf("(%d steres)\n\n", Me->RatsAte);
        printf("Grain\tGrain\tPrice of\tPrice of\tTreasury\n");
        printf("Reserve\tDemand\tGrain\t\tLand\n");
        printf("%d\t%d\t%d\t\t%.2f\t\t%d\n", Me->GrainReserve,
            Me->GrainDemand, Me->GrainPrice, Me->LandPrice,
            Me->Treasury);
        printf("steres\tsteres\t1000 st.\thectare\t\tgold florins\n");
        printf("\nYou have %d hectares of land.\n", Me->Land);
        printf("\n1. Buy grain, 2. Sell grain, 3. Buy land,");
        printf(" 4. Sell land\n(Enter q to continue): ");
        fgets(string, 255, stdin);
        if(string[0] == 'q')
            Finished = True;
        if(string[0] == '1')
            BuyGrain(Me);
        if(string[0] == '2')
            SellGrain(Me);
        if(string[0] == '3')
            BuyLand(Me);
        if(string[0] == '4')
            SellLand(Me);
    }
    return;
}
```



```
void AdjustTax(player *Me)
{
    char string[256];
    int val = 1, duty = 0;
    string[0] = '\0';
    while(val != 0 || string[0] != 'q')
    {
        printf("\n%s %s\n\n", Me->Title, Me->Name);
GenerateIncome(Me);
        printf("(%d%%)\t\t(%d%%)\t\t(%d%%)",
                Me->CustomsDuty, Me->SalesTax,
                Me->IncomeTax);
    printf("\n1. Customs Duty, 2. Sales Tax, 3. Wealth Tax, ");
        printf("4. Justice\n");
        printf("Enter tax number for changes, q to continue: ");
        fgets(string, 255, stdin);
        val = (int)atoi(string);
        switch(val)
        {
            case 1: printf("New customs duty (0 to 100): ");
                fgets(string, 255, stdin);
                duty = (int)atoi(string);
                if(duty > 100) duty = 100;
                if(duty < 0) duty = 0;
                Me->CustomsDuty = duty;
                break;
            case 2: printf("New sales tax (0 to 50): ");
                fgets(string, 255, stdin);
                duty = (int)atoi(string);
                if(duty > 50) duty = 50;
                if(duty < 0) duty = 0;
                Me->SalesTax = duty;
                break;
            case 3: printf("New wealth tax (0 to 25): ");
                fgets(string, 255, stdin);
                duty = (int)atoi(string);
                if(duty > 25) duty = 25;
                if(duty < 0) duty = 0;
                Me->IncomeTax = duty;
                break;
            case 4: printf("Justice: 1. Very fair, 2. Moderate");
                printf(" 3. Harsh, 4. Outrageous: ");
                fgets(string, 255, stdin);
                duty = (int)atoi(string);
                if(duty > 4) duty = 4;
                if(duty < 1) duty = 1;
                Me->Justice = duty;
                break;
        }

    }
    AddRevenue(Me);
    if(Me->IsBankrupt == True)
    SeizeAssets(Me);
}
void DrawMap(player *Me)
{
    /* Not implemented yet. */
    return;
}
void StatePurchases(player *Me, int HowMany, player MyPlayers[6])
{
```

```
    char string[256];
    int val = 1;
    string[0] = '\0';
    while(val != 0 || string[0] != 'q')
    {
    printf("\n\n%s %s\nState purchases.\n", Me->Title, Me->Name);
        printf("\n1. Marketplace (%d)\t\t\t1000 florins\n",
                Me->Marketplaces);
        printf("2. Woolen mill (%d)\t\t\t2000 florins\n",
                Me->Mills);
        printf("3. Palace (partial) (%d)\t\t\t3000 florins\n",
                Me->Palace);
        printf("4. Cathedral (partial) (%d)\t\t\t5000 florins\n",
                Me->Cathedral);
    printf("5. Equip one platoon of serfs as soldiers\t500 florins\n");
        printf("\nYou have %d gold florins.\n", Me->Treasury);
        printf("\nTo continue, enter q. To compare standings, enter 6\n");
        printf("Your choice: ");
        fgets(string, 255, stdin);
        val = (int)atoi(string);
        switch(val)
        {
            case 1: BuyMarket(Me); break;
            case 2: BuyMill(Me); break;
            case 3: BuyPalace(Me); break;
            case 4: BuyCathedral(Me); break;
            case 5: BuySoldiers(Me); break;
            case 6: ShowStats(MyPlayers, HowMany);
        }
    }
    return;
}
void ShowStats(player MyPlayers[6], int HowMany)
{
    int i = 0;
    char string[256];
    printf("Nobles\tSoldiers\tClergy\tMerchants\tSerfs\tLand\tTreasury\n");
    for(; i < HowMany; i++)
        printf("\n%s %s\n%d\t%d\t\t%d\t\t%d\t\t%d\t%d\t\t%d\n",
                MyPlayers[i].Title, MyPlayers[i].Name,
                MyPlayers[i].Nobles, MyPlayers[i].Soldiers,
                MyPlayers[i].Clergy, MyPlayers[i].Merchants,
                MyPlayers[i].Serfs, MyPlayers[i].Land,
                MyPlayers[i].Treasury);
    printf("\n(Press ENTER): ");
    fgets(string, 255, stdin);
    return;
}
void ImDead(player *Me)
{
    char string[256];
    int why;

    printf("\n\nVery sad news.\n%s %s has just died\n", Me->Title,
            Me->Name);

    if(Me->Year > 1450)
        printf("of old age after a long reign.\n");
    else
    {
        why = Random(8);
        switch(why)
```



```
        {
            case 0:
            case 1:
            case 2:
            case 3: printf("of pneumonia after a cold winter in a drafty castle.\n"); break;
            case 4: printf("of typhoid after drinking contaminated water.\n"); break;
            case 5: printf("in a smallpox epidemic.\n"); break;
            case 6: printf("after being attacked by robbers while travelling.\n"); break;
            case 7:
            case 8: printf("of food poisoning.\n"); break;
        }
    }
    Me->IsDead = True;
    printf("\n(Press ENTER): ");
    fgets(string, 255, stdin);
    return;
}
```