// Description: This program creates a budget calculator

#include <stdio.h>

/\* function prototypes \*/

/\* This function ask if it is a leap year \*/

void printMonth (int \*pMonth)

{

switch (\*pMonth)

{

case 1: printf ("January "); break;

case 2: printf ("February "); break;

case 3: printf ("March "); break;

case 4: printf ("April "); break;

case 5: printf ("May "); break;

case 6: printf ("June "); break;

case 7: printf ("July "); break;

case 8: printf ("August "); break;

case 9: printf ("September "); break;

case 10: printf ("October "); break;

case 11: printf ("November "); break;

case 12: printf ("December "); break;

}

}

void printDay (char \*pDay)

{

switch (\*pDay)

{

case 'S':

case 's': printf ("Sunday "); break;

case 'a':

case 'A': printf ("Saturday "); break;

case 'm':

case 'M': printf ("Monday "); break;

case 't':

case 'T': printf ("Tuesday "); break;

case 'w':

case 'W': printf ("Wednesday "); break;

case 'h':

case 'H': printf ("Thursday "); break;

case 'f':

case 'F': printf ("Friday "); break;

}

}

void logIncome (float \*pBudget, float \*CurrentIncome)

{

float fIncome;

do

{

printf ("Enter income amount you like to log: ");

scanf ("%f", &fIncome);

if (fIncome < 0)

printf ("Income amount cannot be negative.\n\n");

else

{

\*pBudget = \*pBudget + fIncome;

\*CurrentIncome = \*CurrentIncome + fIncome;

printf ("Total Daily Income Today: PHP %.2f", \*CurrentIncome);

printf ("\nCurrent Budget: PHP %.2f", \*pBudget);

}

} while (fIncome < 0);

}

void logExpense (float \*pBudget, float \*CurrentExpense)

{

float fExpense;

do

{

printf ("Enter expense amount you like to log: \n");

scanf ("%f", &fExpense);

if (fExpense < 0)

printf ("Expense amount cannot be negative\n\n");

} while (fExpense < 0);

\*pBudget = \*pBudget - fExpense;

\*CurrentExpense = \*CurrentExpense + fExpense;

printf ("\nTotal Daily Expense Today: %.2f", \*CurrentExpense);

printf ("\nCurrent Budget: PHP %.2f", \*pBudget);

}

void scheduleRepeatingIncome (int \*pRepeatingIncomeDate, float \*pRepeatingIncomeAmount, int \*pIncomeChecker)

{

int nDate;

float fRepeatingIncome;

do

{

printf ("Enter date of repeating income: ");

scanf ("%d", &nDate);

if (nDate < 0 || nDate > 31)

printf ("Invalid Date! Please try again. ^^\n");

else

{

printf ("Enter the repeating income amount: ");

scanf ("%f", &fRepeatingIncome);

if (fRepeatingIncome < 0)

printf ("Repeating Income cannot be negative. Please input again. ^^\n");

else

{

\*pRepeatingIncomeDate = nDate;

\*pRepeatingIncomeAmount = fRepeatingIncome;

printf ("The amount of %.2f will be added to the budget every day %d of the month", fRepeatingIncome, nDate);

\*pIncomeChecker = 1;

}

}

} while (nDate < 0 || nDate > 31 || fRepeatingIncome < 0);

}

void updateRepeatingIncome (int \*pRepeatingIncomeDate, float \*pRepeatingIncomeAmount, int \*pIncomeChecker)

{

char cUpdateChecker;

char cRemovePrev;

int nDate;

float fRepeatingIncome;

do

{

printf ("Are you sure you want to update your repeating income? <Y/N> \n");

scanf (" %c", &cUpdateChecker);

if (cUpdateChecker == 'Y' || cUpdateChecker == 'y')

{

do

{

printf ("Do you wish to remove your previous repeating income? [Y/N] \n");

scanf (" %c",&cRemovePrev);

if (cRemovePrev == 'Y' || cRemovePrev == 'y')

{

\*pIncomeChecker = 0;

\*pRepeatingIncomeDate = 0;

\*pRepeatingIncomeAmount = 0;

printf ("Previous repeating expense is removed.\n");

}

else

{

do

{

printf ("Enter new date of repeating income: ");

scanf ("%d", &nDate);

if (nDate < 0 || nDate > 31)

printf ("Invalid Date! Please try again. ^^\n");

else

{

printf ("Enter the new repeating income amount: ");

scanf ("%f", &fRepeatingIncome);

if (fRepeatingIncome < 0)

printf ("Repeating Income cannot be negative. Please input again. ^^\n");

else

{

\*pRepeatingIncomeDate = nDate;

\*pRepeatingIncomeAmount = fRepeatingIncome;

printf ("The amount of %.2f will be added to the budget every day %d of the month", fRepeatingIncome, nDate);

\*pIncomeChecker = 1;

}

}

} while (nDate < 0 || nDate > 31 || fRepeatingIncome < 0);

}

} while (cRemovePrev != 'Y' || cRemovePrev != 'y' || cRemovePrev != 'N' || cRemovePrev != 'n');

}

else

{

\*pIncomeChecker = 1;

}

} while (cUpdateChecker != 'Y' || cUpdateChecker != 'y' || cUpdateChecker != 'N' || cUpdateChecker != 'n');

}

void scheduleRepeatingExpense (float \*pBudget, int \*pRepeatingExpenseDate, float \*pRepeatingExpenseAmount, int \*pExpenseChecker, float \*pPayables)

{

int nDate;

float fRepeatingExpense;

do

{

printf ("Enter date of repeating expense: ");

scanf ("%d", &nDate);

if (nDate < 0 || nDate > 31)

printf ("Invalid Date! Please try again. ^^\n");

else

{

printf ("Enter the repeating expense amount: ");

scanf ("%f", &fRepeatingExpense);

if (fRepeatingExpense < 0)

printf ("Repeating Expense cannot be negative. Please try again. ^^\n");

else if (\*pBudget < fRepeatingExpense)

{

printf ("%.2f can't be deducted from the budget due to insufficient balance, instead the %.2f will be added to the payables.\n", fRepeatingExpense, fRepeatingExpense);

\*pPayables += fRepeatingExpense;

\*pExpenseChecker = 1;

}

else

{

\*pRepeatingExpenseDate = nDate;

\*pRepeatingExpenseAmount = fRepeatingExpense;

printf ("The amount of %.2f will be deducted to the budget every day %d of the month", fRepeatingExpense, nDate);

\*pExpenseChecker = 1;

}

}

} while (nDate < 0 || nDate > 31 || fRepeatingExpense < 0);

}

void updateRepeatingExpense (float \*pBudget, int \*pRepeatingExpenseDate, float \*pRepeatingExpenseAmount, int \*pExpenseChecker, float \*pPayables)

{

char cUpdateChecker;

char cRemovePrev;

int nDate;

float fRepeatingExpense;

do

{

printf ("Are you sure you want to update your repeating expense? <Y/N> ");

scanf (" %c", &cUpdateChecker);

if (cUpdateChecker == 'Y' || cUpdateChecker == 'y')

{

do

{

printf ("Do you wish to remove your previous repeating expense? <Y/N> \n");

scanf (" %c",&cRemovePrev);

if (cRemovePrev == 'Y' || cRemovePrev == 'y')

{

\*pExpenseChecker = 0;

\*pRepeatingExpenseDate = 0;

\*pRepeatingExpenseAmount = 0;

printf ("Previous repeating expense is removed.\n");

}

else

{

do

{

printf ("Enter date of repeating expense: ");

scanf ("%d", &nDate);

if (nDate < 0 || nDate > 31)

printf ("Invalid Date! Please try again. ^^\n");

else

{

printf ("Enter the repeating expense amount: ");

scanf ("%f", &fRepeatingExpense);

if (fRepeatingExpense < 0)

printf ("Repeating Expense cannot be negative. Please try again. ^^\n");

else if (\*pBudget < fRepeatingExpense)

{

printf ("%.2f can't be deducted from the budget due to insufficient balance, instead the %.2f will be added to the payables.\n", fRepeatingExpense, fRepeatingExpense);

\*pPayables += fRepeatingExpense;

\*pExpenseChecker = 1;

}

else

{

\*pRepeatingExpenseDate = nDate;

\*pRepeatingExpenseAmount = fRepeatingExpense;

printf ("The amount of %.2f will be deducted to the budget every day %d of the month", fRepeatingExpense, nDate);

\*pExpenseChecker = 1;

}

}

} while (nDate < 0 || nDate > 31 || fRepeatingExpense < 0);

}

} while (cRemovePrev != 'Y' || cRemovePrev != 'y' || cRemovePrev != 'N' || cRemovePrev != 'n');

}

else

{

\*pExpenseChecker = 1;

}

} while (cUpdateChecker != 'Y' || cUpdateChecker != 'y' || cUpdateChecker != 'N' || cUpdateChecker != 'n' || \*pExpenseChecker == 0);

}

void scheduleRentExpense (float \*pBudget, float \*pRentExpenseAmount, int \*pRentChecker, float \*pPayables)

{

float fRentExpense;

printf ("Enter amount of monthly rent expense: ");

scanf ("%f", &fRentExpense);

if (\*pBudget < fRentExpense)

{

printf ("%.2f can't be deducted from the budget due to insufficient balance, instead the %.2f will be added to the payables.\n", fRentExpense, fRentExpense);

\*pPayables += fRentExpense;

\*pRentChecker = 1;

}

else

{

\*pRentExpenseAmount = fRentExpense;

printf ("The amount of %.2f will be repeatedly deducted to the budget every end of the month\n",fRentExpense);

\*pRentChecker = 1;

}

}

void updateRentExpense (float \*pBudget, float \*pRentExpenseAmount, int \*pRentChecker, float \*pPayables)

{

char cUpdateChecker;

char cRemovePrev;

float fRentExpense;

do

{

printf ("Are you sure you want to update your rent expense? <Y/N> ");

scanf (" %c", &cUpdateChecker);

if (cUpdateChecker == 'Y' || cUpdateChecker == 'y')

{

do

{

printf ("Do you wish to remove your previous rent expense? <Y/N> \n");

scanf (" %c",&cRemovePrev);

if (cRemovePrev == 'Y' || cRemovePrev == 'y')

{

\*pRentChecker = 0;

\*pRentExpenseAmount = 0;

printf ("Previous rent expense is removed.\n");

}

else

{

printf ("Enter amount of monthly rent expense: ");

scanf ("%f", &fRentExpense);

if (\*pBudget < fRentExpense)

{

printf ("%.2f can't be deducted from the budget due to insufficient balance, instead the %.2f will be added to the payables.\n", fRentExpense, fRentExpense);

\*pPayables += fRentExpense;

\*pRentChecker = 1;

}

else

{

\*pRentExpenseAmount = fRentExpense;

printf ("The amount of %.2f will be repeatedly deducted to the budget every end of the month\n",fRentExpense);

\*pRentChecker = 1;

}

}

} while (cRemovePrev != 'Y' || cRemovePrev != 'y' || cRemovePrev != 'N' || cRemovePrev != 'n');

}

else

{

\*pRentChecker = 1;

}

} while (cUpdateChecker != 'Y' || cUpdateChecker != 'y' || cUpdateChecker != 'N' || cUpdateChecker != 'n');

}

void EndDay (char \*pDay)

{

int nDayCount;

printf ("\n✧✧✧✧✧✧✧✧✧✧ END OF DAY ✧✧✧✧✧✧✧✧✧✧\n\n\n\n");

printf ("✧✧✧✧✧✧✧✧ START OF THE DAY ✧✧✧✧✧✧✧✧\n");

switch (\*pDay)

{

case 'a':

case 'A':

nDayCount = 1; break;

case 's':

case 'S':

nDayCount = 2; break;

case 'm':

case 'M':

nDayCount = 3; break;

case 't':

case 'T':

nDayCount = 4; break;

case 'w':

case 'W':

nDayCount = 5; break;

case 'h':

case 'H':

nDayCount = 6; break;

case 'f':

case 'F':

nDayCount = 7; break;

}

nDayCount++;

if (nDayCount == 8)

nDayCount = 1;

printf ("Today is ");

switch (nDayCount)

{

case 1: printf ("Saturday "); break;

case 2: printf ("Sunday "); break;

case 3: printf ("Monday "); break;

case 4: printf ("Tuesday "); break;

case 5: printf ("Wednesday "); break;

case 6: printf ("Thursday "); break;

case 7: printf ("Friday "); break;

}

// if (nMonth == 2 && nDate == 2 )

}

// void SkipEndWeek ()

// GET INFORMATION AND MENU

void getInfo (int nMonth, int nDate, char cLeap, char cDay, float fBudget)

{

int nValidLeap = 0, nValidDate = 0;

cLeap = 0;

while (cLeap != 'Y' && cLeap != 'y' && cLeap != 'N' && cLeap != 'n')

{

printf ("\nIs this year a leap year? <Y/N> ");

scanf (" %c", &cLeap);

if (cLeap == 'Y' || cLeap == 'y')

nValidLeap = 0; // A LEAP YEAR

else if (cLeap == 'N' || cLeap == 'n')

nValidLeap = 1; // NOT A LEAP YEAR

else

printf ("Invalid input!\n");

}

// Current Month

do

{

printf ("\nWhat month is it? Choose from below: ");

printf ("\n 1 - January\n 2 - February\n 3 - March\n 4 - April\n 5 - May\n 6 - June\n 7 - July\n 8 - August\n 9 - September\n 10 - October\n 11 - November\n 12 - December\n");

scanf ("%d", &nMonth);

} while (nMonth < 1 || nMonth > 12);

// Current Date

do

{

printf ("\nWhat date is it?: ");

scanf ("%d", &nDate);

if ((nDate >= 1 && nDate <= 31) && (nMonth == 1 || nMonth == 3 || nMonth == 5 || nMonth == 7 || nMonth == 8 || nMonth == 10 || nMonth == 12))

nValidDate = 0;

else if ((nDate >= 1 && nDate <= 30) && (nMonth == 4 || nMonth == 6 || nMonth == 9 || nMonth == 11))

nValidDate = 0;

else if (nDate >= 1 && nDate <= 28 && nMonth == 2 && nValidLeap == 0) // a leap year

nValidDate = 0;

else if (nDate >= 1 && nDate <= 28 && nMonth == 2 && nValidLeap == 1) // not a leap year so until feb 27

nValidDate = 0;

/\* else if (nMonth == 2 && nDate == 29 && nValidLeap == 1) // not a leap year

{

printf ("Invalid date!\n");

nValidDate = 1;

} \*/

else

{

printf ("Invalid Date!\n");

nValidDate = 1;

}

} while (nDate < 1 || nDate > 31 || nValidDate == 1);

// Current Day

do

{

printf ("\nWhat day is it? Choose from below ");

printf ("\n A - Saturday\n S - Sunday\n M - Monday\n T - Tuesday\n W - Wednesday\n H - Thursday\n F - Friday\n");

scanf (" %c", &cDay);

} while (cDay != 'A' && cDay != 'a' && cDay != 'S' && cDay != 's' && cDay != 'M' && cDay != 'm' && cDay != 'T' && cDay != 't' && cDay != 'W' && cDay != 'w' && cDay != 'H' && cDay != 'h' && cDay != 'F' && cDay != 'f');

// Starting Budget

printf ("\nHow much is your starting budget? \n");

scanf ("%f", &fBudget);

// DISPLAY

printf ("\n✧✧✧✧✧✧✧✧ START OF THE DAY ✧✧✧✧✧✧✧✧\n");

printf ("Today is ");

printDay (&cDay); printMonth (&nMonth); printf ("%d\n", nDate);

if (nValidLeap == 0)

printf ("This year is a leap year!\n");

else

printf ("This year is not a leap year!\n");

printf ("Your current budget is PHP %.2f ", fBudget);

printf ("\n✧✧✧✧✧✧✧✧✧✧✧✧✧✧✧✧✧✧✧✧✧✧✧✧✧✧✧✧✧✧✧✧✧✧\n");

int nChoice;

int nIncomeChecker = 0;

int nExpenseChecker = 0;

int nRentChecker = 0;

int nRepeatingIncomeDate;

int nRepeatingExpenseDate;

float fIncomeAmount = 0;

float fExpenseAmount = 0;

float fRentExpenseAmount = 0;

float fRepeatingIncomeAmount;

float fRepeatingExpenseAmount;

float fPayables;

do

{

printf ("\n\nSelect below:\n");

printf ("1 - Log Income\n");

printf ("2 - Log Expense\n");

if (nIncomeChecker == 0)

printf ("3 - Schedule repeating income\n");

else

printf ("3 - Update repeating income\n");

if (nExpenseChecker == 0)

printf ("4 - Schedule repeating expense\n");

else

printf ("4 - Update repeating expense\n");

if (nRentChecker == 0)

printf ("5 - Set repeating rent expense\n");

else

printf ("5 - Update repeating rent expense\n");

printf ("6 - Display total daily income\n");

printf ("7 - Display total daily expense\n");

printf ("8 - Display current budget\n");

printf ("9 - Display payables\n");

printf ("10 - End the day\n");

printf ("11 - Skip to the end of the week\n");

printf ("12 - Skip to the end of the month\n");

printf ("13 - Skip to the end of the year\n");

printf ("14 - Skip to the specified date\n");

printf ("15 - Exit\n");

// INPUT OPTION

printf ("\nPlease input option: \n");

scanf ("%d", &nChoice);

if (nChoice == 1)

logIncome (&fBudget, &fIncomeAmount);

else if (nChoice == 2)

logExpense (&fBudget, &fExpenseAmount);

else if (nChoice == 3)

{

if (nIncomeChecker == 0)

scheduleRepeatingIncome (&nRepeatingIncomeDate, &fRepeatingIncomeAmount, &nIncomeChecker);

else

updateRepeatingIncome (&nRepeatingIncomeDate, &fRepeatingIncomeAmount, &nIncomeChecker);

}

else if (nChoice == 4)

{

if (nExpenseChecker == 0)

scheduleRepeatingExpense (&fBudget, &nRepeatingExpenseDate, &fRepeatingExpenseAmount, &nExpenseChecker, &fPayables);

else

updateRepeatingExpense (&fBudget, &nRepeatingExpenseDate, &fRepeatingExpenseAmount, &nExpenseChecker, &fPayables);

}

else if (nChoice == 5)

{

if (nRentChecker == 0)

scheduleRentExpense (&fBudget, &fRentExpenseAmount, &nRentChecker, &fPayables);

else

updateRentExpense (&fBudget, &fRentExpenseAmount, &nRentChecker, &fPayables);

}

else if (nChoice == 6)

{

float fDisplay;

fDisplay = fIncomeAmount;

if (nDate == nRepeatingIncomeDate)

fDisplay += fRepeatingIncomeAmount;

printf ("Today's Total Daily Income: %.2f\n", fDisplay);

}

else if (nChoice == 7)

{

float fDisplay;

fDisplay = fExpenseAmount;

if (nDate == nRepeatingExpenseDate)

fDisplay += fRepeatingExpenseAmount;

if (nDate == 28 || nDate == 29|| nDate == 30 || nDate == 31)

fDisplay += fRentExpenseAmount;

printf ("Today's Total Daily Expense: %.2f\n", fDisplay);

}

else if (nChoice == 8)

{

printf ("Current Budget: %.2f\n", fBudget);

}

else if (nChoice == 9)

{

if (fPayables <= 0)

printf ("You have no payables as of today. ^^\n");

else

printf ("Payables: %.2f\n", fPayables);

}

else if (nChoice == 10)

EndDay (&cDay);

/\*

else if (nChoice == 11)

SkipEndWeek ();

else if (nChoice == 12)

SkipEndMonth ();

else if (nChoice == 13)

SkipEndYear ();

\*/

else

printf ("See you next time!\n");

} while (nChoice != 15);

}

int main ()

{

int nMonth = 0, nDate = 0;

char cLeap = 0, cDay = 0;

float fStartingBudget = 0.0;

getInfo (nMonth, nDate, cLeap, cDay, fStartingBudget);

return 0;

}