UNIVERSITEIT·STELLENBOSCH·UNIVERSITY
jou kennisvennoot • your knowledge partner

Computer Programming 143

Practical 6- Memo

2016

Assignment 6A

```c
/* Filename: Assignment6A.c
* Date: 2016/01/01
* Name: Doe J.J.
* Student number: 12345678
* ------------------------------------------------------
* By submitting this file electronically, I declare that
* it is my own original work, and that I have not copied
* any part of it from another source.
* ------------------------------------------------------
* This program introduces different ways to use pointers
* ------------------------------------------------------
*/

#include <stdio.h>

int main(void)
{
        int i;  // obtain memory location that will hold an integer
        int *iPtr;      // obtain memory location that will hold the address of a memory location that holds an integer

        i = 5;  // assign a value to memory location i
        // display value of memory location using its label
        printf("\nThe value stored in the memory location"
                        "labelled i is %d", i);
        printf("\nAddress of memory location"
                        "labelled i is %ld", (long)&i);
        printf("\n\n=============================================================");

        printf("\n\nREFERRING TO i THROUGH INDIRECT MEANS:");
        iPtr = &i;      // assign address of i to pointer iPtr
        printf("\nThe value stored in the memory location"
                        "labelled i is %d", *iPtr);
        printf("\nAddress of memory location"
                        "labelled i is %ld", (long)iPtr);
        printf("\n\n=============================================================");

        printf("\n\nCHANGING i THROUGH INDIRECT MEANS");
        *iPtr = 100;
        printf("\nValue in memory labelled i has changed"
                        "to %d\n(accessed with label)",i);
        printf("\nValue in memory labelled i has changed"
                        "to %d\n(accessed with pointer)",*iPtr);

        printf("\n\nCALCULATIONS WITH i THROUGH INDIRECT MEANS:");
        *iPtr = *iPtr *10;
        printf("\nValue in memory labelled i has been multiplied"
                        "by 10 to give %d\n(accessed with label)",i);
        printf("\nValue in memory labelled i has been multiplied"
                        "by 10 to give %d\n(accessed with pointer)",*iPtr);

        return 0;
}
```

Assignment 6A

main.c

```c
/*
 /* Filename: Assignment6B.c
* Date: 2016/01/01
* Name: Doe J.J.
* Student number: 12345678
* ------------------------------------------------------
* By submitting this file electronically, I declare that
* it is my own original work, and that I have not copied
* any part of it from another source.
* ------------------------------------------------------
* This file contains the main function for calculating
* the trajectory of a projectile.
* ------------------------------------------------------
*/
#include <stdio.h>
#include <stdlib.h>
#include "Assignment6B.h"

int main(void)
{
        char displayData[WIDTH][HEIGHT]; // Our display array
        double trajData[MAX_POINTS][2]; // Our trajectory array
        int a; // Just a dummy input variable
        double speedInit, launchAngle; // Local input variables

        setbuf(stdout, 0);

        do{
                printf("Projectile Trajectory Display:\n");
                printf("Launch Speed (m/sec)?"); scanf("%lf",&speedInit); // Add code to limit user input
                printf("Launch Angle (deg)?"); scanf("%lf",&launchAngle); // Add code to limit user input

                clearDisplayData(displayData); // Clear the display
                // To check that our display works
                // printData(displayData); // print the cleared display
                // Calculate the trajectory
                calcTraj(speedInit, launchAngle, trajData);// Smoother function

                // Place the trajectory data on the display and print it
                updateDisplayData(displayData, trajData);

                printf("Projectile Trajectory Display for V_i = %4.1f m/sec,
                                                Angle_i = %4.1f deg\n\n", speedInit, launchAngle);
                printData(displayData); // Print the trajectory

                printf("Type a 0 to exit!");
                scanf("%d",&a);
        } while (a != 0);

        return 0;
}
```

Assignment6B.h

```c
/*
 /* Filename: Assignment6B.h
* Date: 2016/01/01
* Name: Doe J.J.
* Student number: 12345678
* -----------------------------------------------------
* By submitting this file electronically, I declare that
* it is my own original work, and that I have not copied
* any part of it from another source.
* -----------------------------------------------------
* This file contains all the function prototypes related
* to the calculation of the trajectory of a projectile
* and constants which will be used
* -----------------------------------------------------
*/
#ifndef ASSIGNMENT6B_H_
#define ASSIGNMENT6B_H_

#include <stdio.h>
#include <stdlib.h>
#include <math.h>

// Constants
#define WIDTH 80 // Display size
#define HEIGHT 40
#define MAX_POINTS 300
#define GRAVITY 9.81 // Gravity acceleration
```

```c
#define HALF 0.5
#define FZERO 0.0

// Prototypes
void calcTraj(double speedInit, double launchAngle, double trajData[][2]);
void clearDisplayData(char displayData[][HEIGHT]);
void updateDisplayData(char displayData[][HEIGHT], double trajData[][2]);
void printData(char displayData[][HEIGHT]);

#endif /* ASSIGNMENT6B_H_ */
```

Assignment6B.c

```c
/* Filename: Assignment6B.c
* Date: 2016/01/01
* Name: Doe J.J.
* Student number: 12345678
* ------------------------------------------------------
* By submitting this file electronically, I declare that
* it is my own original work, and that I have not copied
* any part of it from another source.
* ------------------------------------------------------
* This file contains all the functions related to the
* calculation of the trajectory of a projectile.
* ------------------------------------------------------
*/
#include "Assignment6B.h"

/* ********************************************************************
 * Function: clearDisplayData
 * Purpose: Clear the display area with spaces and add border
 * Inputs: Display area (address of array)
 * Outputs: None
 * Operation: Fill display area with spaces, then add border characters
 *
 **********************************************************************/
void clearDisplayData(char displayData[][HEIGHT])
{
        int x,y;
        // Clear the display
        for (x = 0; x < WIDTH; x++){
                for (y = 0; y < HEIGHT; y++){
                        displayData[x][y] = ' '; // Insert Space
                }
        }
        // Draw the border
        for (x = 1; x < (WIDTH - 1); x++){
                displayData[x][0] = '-'; // Top border
                displayData[x][HEIGHT-1] = '-'; // Bottom border
        }
        for (y = 0; y < HEIGHT; y++){
                displayData[0][y] = '|'; // Left border
                displayData[WIDTH-1][y] = '|'; // Right border
        }
}

/* ********************************************************************
 * Function: printData
 * Purpose: Print the display data on the screen
 * Inputs: Display data (address of array)
 * Outputs: None
 * Operation: Print the display
 *
 **********************************************************************/
void printData(char displayData[][HEIGHT])
{
        int x,y;
        // Print the display
        for (y = 0; y < HEIGHT; y++){
                for (x = 0; x < WIDTH; x++){
                        // Print character
                        putchar(displayData[x][y]); // Alternative printf("%c",displayData[x][y]);
                }
                putchar(’\n’); // printf("\n"); Next line
        }
}

/* ********************************************************************
 * Function: updateDisplayData
 * Purpose: Draw the trajectory onto the display
 * Inputs: Initial speed (m/sec) and launch angle (in degrees)
 * Outputs: X-Y trajectory data
 * Operation: Calculate the trajectory data
 *
 **********************************************************************/
void updateDisplayData(char displayData[][HEIGHT], double trajData[][2])
{
        int i, xInt, yInt;
        for (i = 0; i < MAX_POINTS; i++){
                yInt = (int) floor(trajData[i][1]); // Get integer value of Y
                if (yInt >= 0){ // Is Y still positive?
                        xInt = (int) floor(trajData[i][0]); // Get integer value of X
                        if ((xInt < WIDTH) && (yInt < HEIGHT)) // Watch for too big indexes
                                displayData[xInt][HEIGHT - 1 - yInt] = ’*’; // Note: Y-axis is flipped
                }
        }
}

```

```c
/* *******************************************************************
 * Function: calcTraj2
 * Purpose: Calculate the trajectory
 * Inputs: Initial speed (m/sec) and launch angle (in degrees)
 * Outputs: X-Y trajectory data
 * Operation: Calculate the trajectory data
 *
 *********************************************************************/
void calcTraj(double speedInit, double launchAngle, double trajData[][2])
{
        int i;
        double angleRad,speedInitX, speedInitY, temp;
        // Init the trajectory array (with no trajectory data)
        for (i = 0; i < MAX_POINTS; i++){
                trajData[i][0] = trajData[i][1] = -1.0; // negative = no trajectory path
        }
        // Get components and (time = 1 distance unit/speed) increment
        angleRad = launchAngle * M_PI / 180.0; // Convert to radians
        speedInitX = speedInit * cos(angleRad); // X-component of launch speed
        speedInitY = speedInit * sin(angleRad); // Y-component of launch speed
        speedInitX = fmax(speedInitX, 0.1); // make sure X-component is more than zero
        // Calculate the trajectory components
        i = 0;
        do {
                trajData[i][0] = i; // Next X - position
                temp = i * (speedInitY - HALF * GRAVITY * i / speedInitX) / speedInitX;
                trajData[i][1] = temp;
                i++; // and index
        } while (temp >= FZERO); // Stop when y = 0 is crossed
}
```