Data Structures and Algorithms
Assignment 1 / **Solution**

Mar 7, 2016

Introduction to C

Task 1

```
#include <stdio.h>

void reverse(char word[], char reverseString[]) {
    int i;
    int wordLength;

    wordLength = 0;
    while(word[wordLength] != '\0')
        wordLength++;
    for (i = 0; word[i] != '\0'; i++)
        reverseString[wordLength - i - 1] = word[i];
    reverseString[wordLength] = '\0';
}

void main() {
    int maxstrlen = 100;
    char strA[maxstrlen];
    char reverseString[maxstrlen];

    printf("Type a string: ");
    scanf("%[^\n]s", strA);

    reverse(strA, reverseString);
    printf("Reverse string: %s\n", reverseString);
}

// Linux, Mac: gcc task1.c -o task1; ./task1
// Windows: gcc task1.c -o task1; task1
```

Task 2

```
#include <stdio.h>
```

```
int countCopies(int a[], int size, int x) {
    int count = 0;
    int i;
    for (i = 0; i < size; i++) {
        if (a[i] == x) {
            count++;
        }
    }
    return count;
}

void main() {
    int maxarraylen = 1000;
    int i, n;
    int x;
    int a[maxarraylen];

    printf("Type elements of A seperated by spaces (non-number to stop)
        : ");
    i=0;
    while(scanf("%d", &a[i]) == 1) i++;
    n=i;
    // Read but do not store any terminating not integer values ('end')
    scanf("%*s");

    printf("Type x: ");
    scanf("%d", &x);

    printf("Copies = %d\n", countCopies(a, n, x));
}

// Linux, Mac: gcc task2.c -o task2; ./task2
// Windows: gcc task2.c -o task2; task2
```

Task 3

```
#include <stdio.h>

void countPairs(int a[], int nA, int b[], int nB) {
    int i = 0, j = 0, count = 0;
    while (i<nA) {
        while (a[i] < b[j] && i < nA) {
            printf("(%d, %d) ", a[i], count);
            i++;
            count=0;
        }
        while (a[i] > b[j] && j < nB) j++;
        while (a[i] == b[j] && j < nB) { j++; count++; }
    }
    printf("\n");
}
```

Sorting (n^2)

Task 4

```
void bubblesort(int a[], int size) {
    int t, i, j;
    for (i = size - 1; i > 0; i--) {
        for (j = 1; j <= i; j++) {
            if (a[j] < a[j-1]) {
                t = a[j];
                a[j] = a[j-1];
                a[j-1] = t;
            }
        }
    }
}

int team_photo(int A[], int B[], int size) {
    int i;
    int isPossible = 1;
    bubblesort(A, size);
    bubblesort(B, size);

    isPossible = A[0] != B[0];
    for (i = 1; i < size && isPossible; i++)
        isPossible = isPossible && (A[i]!=B[i]) && ((A[i]<B[i]) == (A[i
            -1]<B[i-1]));

    return isPossible;
}
```

Recursion

Task 5

```
int power(int base, int exponent) {
    if (exponent == 0) return 1;
    if (exponent % 2) return base*power(base, exponent-1);
    else {
        int temp = power(base, exponent/2);
        return temp*temp;
    }
}
```

Task 6

```
int gcf(int a, int b) {
    int big, small;
    if (a == b) return a;
    if (a < b) {
        big = b;
        small = a;
    } else {
        big = a;
        small = b;
    }
    return gcf(small, big - small);
}
```

Task 7

(a)

```
#include <math.h>

void drawLevyCurve(int iter, double r, int phi) {
    if (iter==0) {
        printf("(%f:%d)\n", r, phi);
    } else {
        drawLevyCurve(iter-1, r * sqrt(0.5), phi-45);
        drawLevyCurve(iter-1, r * sqrt(0.5), phi+45);
    }
}
```

(b)

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


#ifdef __MINGW32__
#include <windows.h>
#endif
#ifdef __APPLE__
#include <GLUT/glut.h>
#else
#include <GL/glut.h>
#endif

static float x, y;
int iteration;

void TurnAndForward(double r, int phi) {
    glVertex2f(x, y);
    glVertex2f(x + r * cos(phi / 180.0f * M_PI),
               y + r * sin(phi / 180.0f * M_PI));
    x = x + r * cos(phi / 180.0f * M_PI);
    y = y + r * sin(phi / 180.0f * M_PI);
}
```

```
void drawLevyCurve(int iter, double r, int phi) {
    if (iter==0) {
        TurnAndForward(r, phi);
    } else {
        drawLevyCurve(iter-1, r * sqrt(0.5), phi-45);
        drawLevyCurve(iter-1, r * sqrt(0.5), phi+45);
    }
}

void display() {
    glClear(GL_COLOR_BUFFER_BIT); //clear all pixels
    glColor3f(1.0,1.0,1.0); // use white color to draw the lines
    glBegin(GL_LINES);

    x=-0.5; y=0.0;
    drawLevyCurve(iteration, 1.0, 0);

    glEnd();
    glFlush();
}

void main(int argc, char *argv[])
{
    printf("Type iteration: ");
    scanf("%d", &iteration);
    glutInit(&argc, argv);
    glutCreateWindow("Levy C");
    glutDisplayFunc(display);
    glutMainLoop();
}

// Linux: gcc task7b.c -lglut -lGL -lGLU -lm -o task7b; ./task7b
// MacOS: gcc -Wno-deprecated task7b.c -o task7b -framework OpenGL -framework
     GLUT; ./task7b
// Windows: gcc task7b.c -o task7b -lfreeglut -lopengl32; task7b
```