Große Hausaufgabe Spezifikation

Ich werde ein Spiel machen. Das Thema von des Spiels ist Nikolaus. Man soll Nikolaus um die Welt führen, so dass jedes Haus besucht wird. Das Ziel ist für den Spieler den kürzesten Weg zu finden und auch Nikolaus am Ende nach Hause zu leiten. Es wird mehrere „Welt“ sein, mit unterschiedliche Schwierigkeiten. Ich werde zwei C-Programme machen.

Die Erste wird ein Hilfsprogramm sein, dieses Programm wird von „Rohdateien“ spielbare Karte machen. Das Rohdaten sieht so aus:

N	M
x_1	y_1
x_2	y_2
…	…
x_i	y_i
von_1	zu_1
von_2	zu_2
…	…
von_j	zu_j

N: Anzahl der Knoten
M: Anzahl der Kanten

x_i: x Koordinate der i-ste Knote
y_i: y Koordinate der i-ste Knote

von_i zu_i: Kante zwischen von_j-ste Knote und zu_j-ste Knote

Der Graph ist ungerichtet und gewichtet. Das Gewicht ist die Entfernung von den Knoten, kalkuliert durch den Positionen.

Spielbare Karte sieht ungefähr gleich, aber dieses Hilfsprogram macht ein neues File, mit der folgenden Extradateien: Die Länge der erste, zweite, dritte kürzeste Weg gerundet zu ganze Zahl. Diese Zahlen werden durch brute-force + Minimumsuchen Algorithmus kalkuliert. (Die kürzesten Hamilton-Kreis)

Die Zweite wird das Mainprogramm sein. Am Anfang wird das Programm das Mainmenu am Fenster anschauen. Dort kann der Spieler wählen, ob welche Karte geladen wird. Man kann auch sehen wie gut er den ausgewählte Karte gemacht. Es ist 4 Möglichkeiten: Best, 2. Best, 3. Best, oder man hat es noch nicht gemacht.

Das aktuelle Spiel sieht aus, so dass der Anfangsknote als Werkstatt von Nikolaus ausgezeichnet wird, und andere Knoten wird als Häuser ausgezeichnet werden. Die Graphics werden mit SDL gelöst. Die Anleitung wird mit Klicken gelöst, also Nikolaus geht da, wo der Spieler klickt, falls es möglich ist. Wenn der Weg ist zu lang, oder Nikolaus ist beschränkt, oder Nikolaus haben nicht den ganzen Graph eingetreten, dann wird das Spiel pausen und informieren den Spieler, dass er kann wiederbeginnen oder ausfahren. Wenn der Spieler kann die Graph regulär herumgehen, dann das Programm kalkuliert, wie gut hat er den Karte gelöst, und wenn er den Karte genug gut gemacht hat, dann spart das Programm die Ergebnisse in einem File. Dieses File wird jeder Lauf geladen, so kann man seine Leistung sehen.

Dieses File sieht so aus:

Map_name_1	Map_score_1
…	…
Map_name_i	Map_score_i

Jede Zeile beinfaltet 2 Datei, die Name der Karte und das Ergebnis bis das Ende der File.