3. Regionalwettbewerb Jena "Jugend forscht"

Ein Computerprogramm zur Unterstützung des mentalen Trainings der in einem Sportverein des Deutschen Schützenbundes e. V. organisierten jugendlichen Sportschützen

Eric Ackermann
Carl-Zeiss-Gymnasium Jena

15. Januar 2016

betreut durch:
Herrn Bert Kunze, Trainer
Herrn Otto Thiele, Informatiklehrer

Kurzfassung

Diese „Jugend forscht“- Arbeit beinhaltet die Entwicklung einer PC-Software zur Unterstützung des Trainings jugendlicher Sportschützen. Das Programm wurde in Zusammenarbeit mit einem Informatiklehrer und dem Trainer eines Thüringer Sportschützenvereins entwickelt.
Insbesondere für jugendliche Sportschützen ergibt sich die Problematik, dass zwar die Schusstechnik im Trainingsverlauf verbessert werden kann, der im Wettbewerb auftretende psychische Druck jedoch häufig zu Fehlverhalten wie Abbruch der Serie führen kann. Eine Visualisierung der im nächsten Schuss im Training zu erreichenden Mindestringzahl könnte hier Entlastung für den Sportschützen bringen.
Die Suche nach einem diesbezüglichen Programm blieb erfolglos.
Deshalb wurde das Programm „Chancenrechner“, eine PC-Software mit realitätsnaher grafischer Darstellung der Zielscheibe, entwickelt. Es gibt den Sportschützen im Training Empfehlungen für die zu schießende Ringzahl beim nächsten Schuss in Form einer Minimalringzahl und einer empfohlenen Ringzahl. Außerdem enthält das Programm einen Timer, der den Sportschützen dabei hilft, ein besseres Zeitgefühl zu entwickeln. Auch für den Trainer stellt das Programm eine Hilfe dar, da es automatisch eine Schussserie im Training auswertet und das Ergebnis protokolliert.
Die Software „Chancenrechner“ wurde im Sportschützenverein „Bürgeler-SG e.V.“ von Trainer und Sportlern erfolgreich getestet.

Inhaltsverzeichnis

Abbildungsverzeichnis

Tabellenverzeichnis

1 Einleitung

1.1 Problembeschreibung

Im Sommer 2015 wurde ich von meinem Informatiklehrer angesprochen, ob ich für einen Trainer eines Thüringer Sportschützenvereins ein Trainingsprogramm entwickeln, installieren und einrichten könne.
Für einen Sportschützen, welcher auf Ringscheiben schießt, kommt es auf Technik und mentale Stärke an. Bei einem Wettkampf ist die Psyche der größte Gegner des Sportschützen. Durch gute Trainingsarbeit beherrscht der Sportschütze seine Technik, aber der psychische Druck ist ungemein höher. Beispielsweise kann durch eine Schussabgabe mit einer niedrigen Ringzahl[1] bei dem Sportler das Gefühl erzeugt werden, er sei nicht mehr in der Lage, eine gute Gesamtringzahl zu schießen. Schon die kleinste Unsicherheit birgt das Risiko, den Sportschützen zu verunsichern und damit das Ergebnis zu verschlechtern oder sogar zum Abbruch der Serie zu führen. Dies ist das Hauptproblem beim Sportschießen „auf Präzision“.
Die Frage, die sich der Sportler nach jedem Schuss stellt, lautet: „Kann ich mein Ziel noch erreichen, und welche Ringzahl muss ich noch mindestens erzielen?“
Natürlich könnte ein Sportschütze die Frage umgehen, indem er bei jedem Schuss versucht, die Optimalzahl von 10 Ringen zu schießen. Jedoch würde dies ihn unter viel zu hohen Druck mit entsprechendem Fehlerrisiko setzen, zumal auch noch der Zeitfaktor eine nicht zu unterschätzende Rolle spielt. Zusammenfassend ist es in entsprechenden Situationen oft günstiger, zum Beispiel eine schnelle und saubere 8 zu schießen als eine 10 weit zu verfehlen.
In der Realität kann sich folgende Situation ergeben: Ein Sportschütze muss, um die aktuelle Serie zu gewinnen, den derzeit führenden Konkurrenten (84 Ringe) mit mindestens 85 Ringen übertreffen, d.h. die Summe seiner Treffer soll mindestens 85 sein. Mit 9 mal Ringzahl 10 wäre das Ziel erreicht. Dies ist in der Praxis allerdings unrealistisch. Im Beispiel schießt er mit seinen ersten beiden Schüssen folgendermaßen:

[1] Der Wert des aktuellen Schusses wird in Ringen gemessen. Die Zielscheibe, auf die geschossen wird, besteht aus 10 Kreisen, die immer den gleichen Abstand voneinander haben, den sogenannten Ringen. Trifft ein Schuss die Scheibe nicht, wird er mit 0 Ringen gewertet, trifft er zwischen dem äußersten und dem zweitäußersten Ring, wird er als 1 gewertet usw.

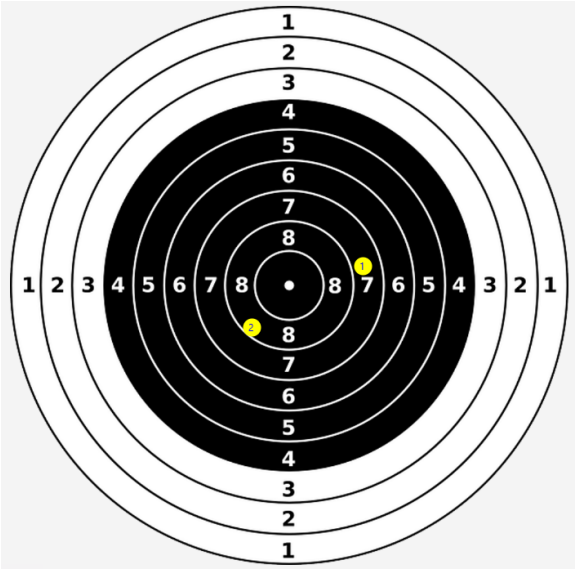

Abbildung 1: Beispielschussfolge eines Sportschützen

Dies ist kein optimaler Start. Es stellt sich die Frage, ob das gesetzte Ziel überhaupt noch zu erreichen ist. Der Stresspegel steigt, der Sportschütze wird nervös, er zittert, Atmung und Herzfrequenz beschleunigen sich. Dadurch wird die Zielgenauigkeit noch weiter beeinträchtigt. Diese Situation kann mit einigen Berechnungen vermieden werden. Um in diesem Beispiel die 85 Ringe zu erreichen, genügt ein Durchschnitt von 8,5 Ringen pro Schuss. Im Beispiel sind mit 8 Schuss noch 70 Ringe zu treffen. Dividiert man nun 70 durch 8, erhält man aufgerundet 9. Daraus ergibt sich die Empfehlung, dass für die nächsten 8 Schüsse jedes Mal 9 Ringe genügen beziehungsweise nach einem folgenden 10er Treffer auch eine Ringzahl von 8 genügt. Die mentale Schwierigkeit für einen Sportschützen im Wettkampf besteht also darin, diese Überlegungen ohne Taschenrechner unter Zeitdruck bei jedem Schuss neu auszuführen.

Auf dem Schießstand des Sportschützenvereins steht permanent ein Windows-Notebook zur Verfügung.
Die Suche nach einem geeigneten Programm, welches Sportschützen bei diesen Überlegungen entlastet, verlief erfolglos. Weder für Windows-PCs noch für Android-Smartphones ließ sich eine adäquate Software finden.
Dadurch entstand die Idee, ein solches Programm selbst zu entwickeln, woraus schließlich diese Arbeit entstand.

1.2 Zielsetzung

Die Software muss zu dem vorhandenen Notebook kompatibel sein. Eine Präsentationsmöglichkeit vor allen Sportschützen ergibt sich aus der Möglichkeit, zusätzlich einen Beamer zu nutzen.
Das Programm soll, während der Sportschütze im Training (im Wettkampf sind Hilfsmittel nicht zugelassen) unter Wettkampfbedingungen schießt, für ihn nützliche Informationen ausgeben. Dies wären zum Beispiel die minimal nötige und die empfohlene Ringzahl für den nächsten Schuss, die bisherige Leistung in Form einer aktuellen Gesamtringzahl sowie die Informati-

on, wo genau die letzten Schüsse getroffen haben. Außerdem soll der Sportschütze jederzeit genau wissen, wie viel Zeit ihm für seine nächsten Schüsse noch bleibt.
Ziel ist, dass der Sportschütze durch das Training mit dem Programm lernt, selbst zu entscheiden, wie viele Ringe er beim nächsten Schuss mit dem für ihn günstigsten Aufwand-Nutzen-Verhältnis erreichen muss. Außerdem soll der Sportschütze das Gefühl entwickeln, dass selbst bei einzelnen nicht zufriedenstellenden Ringzahlen die anvisierte Gesamtringzahl noch erreicht werden kann. Als Nebeneffekt ist zu erwarten, dass ein Nutzer ein entsprechendes Zeitgefühl entwickelt und davon auch im Wettkampf profitieren kann.
Dafür muss das Programm natürlich mit den Wettbewerbsformalismen, zum Beispiel den festen Zuordnungen des Zeitlimits zur Schusszahl oder der Zuordnung der Waffe zur Zielscheibe, kompatibel sein. Es sollten jede beliebige Schuss- und Zielringzahl sowie gebräuchliche Waffen unterstützt werden. Zur Unterstützung des Trainers bietet sich an, ein detailliertes Schussprotokoll zu generieren. Dieses ermöglicht dann, Schwachstellen gemeinsam mit dem Sportschützen zu finden und aktuelle mit älteren Trainingsverläufen zu vergleichen.

2 Dokumentation der Entwicklung des Programms

2.1 Java(FX) als die genutzte Programmiersprache

Java kann in einem Interpreter auf fast jedem Betriebssystem ausgeführt werden, zum Beispiel auf Windows, Linux, MacOS, überall gleich. Es wird auf Milliarden von Geräten bereits ausgeführt und ist kostenlos und leicht zu installieren. Ein Java-Programm muss für alle Systeme nur einmal entwickelt werden und gewährleistet außerdem eine sehr hohe Sicherheit der Programmabläufe.
JavaFX als eine Spezifikation der bekannten Programmiersprache Java ist in der aktuellen Version besonders auf grafische Oberflächen ausgelegt und eignet sich damit hervorragend für die dynamische Visualisierung entsprechender Inhalte. Es ist als modernste Java-Spezifikation zukunftssicher.

2.2 Dokumentation des Hauptalgorithmus: Berechnung der Schussempfehlungen

2.2.1 Algorithmus: Grundidee

Variablen, die auch im Programm existieren, werden im Folgenden kursiv dargestellt.
Um n Ringe bei m Schüssen zu erzielen, ist der nötige Durchschnittswert (im Algorithmus *optimum* genannt) immer $\lceil \frac{n}{m} \rceil$[2].
Also ist die grundlegende Idee, die der Algorithmus verfolgt, den Sportschützen im Durchschnitt zu diesem Wert zu führen.
Zuerst wird (vor Aufruf des Algorithmus') dem Sportschützen empfohlen, im Durchschnitt diesen Wert zu schießen. Die aktuelle Empfehlung ist also *optimum*. Wird nun ein Schussresultat eingegeben, gibt es drei Möglichkeiten:

1. Der Schuss entspricht *optimum*.
2. Der Schuss entspricht einer höheren...
3. ... oder einer niedrigeren Ringzahl.

Im Fall 1 gibt das Programm weiter die Empfehlung *optimum* aus.
In den Fällen 2 und 3 muss die aktuelle Empfehlung angepasst werden. Wie dies im Detail umgesetzt wurde, zeigen die Hellgrün und die Türkisblau markierten Teile der Struktogramme im Anhang. Die Grundidee ist folgende:
Die Summe zweier aufeinanderfolgender Schussergebnisse muss doppelt so groß sein wie *optimum*, um einen Durchschnitt von *optimum* zu erreichen (arithmetisches Mittel). Also wird die Abweichung vom aktuellen Schuss zu *optimum* ermittelt, die Empfehlung für den nächsten Schuss ist dann *optimum* plus oder minus diese Abweichung. Der Wert, der sich dabei ergibt, kann zwischen 0 und 10 Ringen liegen. Hierbei wird diese Ringzahl als Empfehlung für den

[2] Dieser Quotient muss immer aufgerundet werden, um niemals Empfehlungen zu geben, die zu einer zu geringen Ringzahl führen.

nächsten Schuss ausgegeben. Sollte die theoretisch berechnete Ringzahl kleiner als 0 oder größer als 10 (dies sind keine praktisch erreichbaren Ringzahlen) sein, wird *optimum* neu berechnet und als aktuelle Empfehlung ausgegeben.

2.2.2 Zur Umsetzung des Hauptalgorithmus

Der Hauptalgorithmus ist in einem Abschnitt des Programmcodes, einer sogenannten „Methode“, umgesetzt.
Vereinfachend erklärt besteht diese Umsetzung aus einer Reihe von Fragen, die vom Programm in Abhängigkeit der Eingaben mit „Ja“ oder „Nein“ beantwortet werden müssen.
Jenachdem, ob diese Fragen mit „Ja“ oder „Nein“ beantwortet werden, führt das Programm die für diesen Fall hinterlegte Aktion aus.
Angenommen, die erste Frage im Algorithmus „Wurden weniger Ringe als *optimum* geschossen?“ wird mit „Ja“ beantwortet, werden folgende Aktionen ausgeführt:

1. Die Variable[3] *Guthaben* speichert die Differenz der Ringe, die man erhält, wenn man den vorgeschlagenen Durchschnitt erreicht, und der Zielringzahl. Ist *Guthaben* größer oder gleich der aktuellen Abweichung, wird *Guthaben* um diese verringert. Damit wird realisiert, dass die Abweichung ausgeglichen und der Schuss wie ein Schuss, der *optimum* Ringe getroffen hat, behandelt wird.

2. Ist *Guthaben* kleiner als die Abweichung von *optimum*, dann wird geprüft, ob die aktuelle Ringzahl eine Schussempfehlung war. Falls ja, wird die Ganzzahl mit dem Index der aktuellen Ringzahl im Array[4] *zahl* um 1 erniedrigt und im Wahrheitswert[5] *nurabgearbeitet* gespeichert, dass eine Empfehlung befolgt wurde.

3. Falls nein, wird geprüft, ob noch empfohlen wird, *optimum* Ringe zu schießen. Ist dies gegeben, wird *zahl[optimum]* um 1 erniedrigt, da *optimum* in der im Folgenden generierten Empfehlung enthalten ist.

4. Danach wird die Abweichung des aktuellen Wertes von *optimum* ermittelt und geprüft, ob *optimum* plus diese Empfehlung im Bereich zwischen 0 und 10 Ringen liegt. Falls ja, wird empfohlen, *optimum* plus diese Abweichung Ringe zu schießen, damit die Summe des aktuellen und des nächsten Schusses zweimal *optimum* beträgt.

5. Fällt die zum Ausgleich nötige Zahl aus diesem Intervall, muss *optimum* neu berechnet werden. Dafür werden alle Empfehlungen auf 0 gesetzt und *optimum* als Quotient der noch zu erreichenden Ringzahl und der verbleibenden Schusszahl neu berechnet. Dabei muss *optimum* aufgerundet werden, um nicht zu zu wenigen Ringen als Endergebnis zu führen.

6. *Guthaben* wird als Differenz $übrigeSchüsse * optimum + gesamt - ziel$ neu berechnet.

[3] Eine Variable ist ein Teil des Arbeitsspeichers, in dem Daten gespeichert werden können. Variablen haben immer einen „Typ“, der bestimmt, welche Art von Daten sich in ihr befinden. „Guthaben“ zum Beispiel ist vom Typ Ganzzahl, also ein Speicherbereich, in dem der Computer für eine spätere Verwendung eine ganze Zahl speichern kann.

[4] Ein Array (auch „Reihung“) ist ein Datentyp, der beliebig viele Variablen des gleichen Typs z.B. als „Zahlenfolge“ zusammenfasst. In *zahl* werden alle Empfehlungen, die das Programm noch geben will, gespeichert. Dabei sind die Nummern der Zahlen, aus denen *zahl* besteht, die zu gebenden Empfehlungen von 0 bis 10 und die Zahlen selbst die Information, wie oft das Programm diese Empfehlung noch ausgeben will.

[5] Ein Wahrheitswert ist ein Datentyp, der nur zwei Daten speichern kann: „Ja“ oder „Nein“.

7. Im Wahrheitswert *frischgenullt* wird gespeichert, dass gerade ein neues *optimum* berechnet wurde.

8. Zuletzt wird geprüft, ob *optimum* größer als 10 ist. Falls ja, wird im Wahrheitswert *nicht-mehrzuerreichen* gespeichert, dass man die Zielringzahl nicht mehr erreichen kann, sonst wird im Array *zahl* die Empfehlung gespeichert, bei jedem übrigen Schuss *optimum* Ringe zu schießen.

Im möglichen Fall, dass mehr Ringe als *optimum* geschossen wurden, ist das Vorgehen adäquat.
Im Fall, dass genau *optimum* Ringe geschossen wurden, wird nur die *optimum*-te Zahl von *zahl*, wenn sie größer 0 ist, um 1 erniedrigt.
Nachdem die Fragen „Wurden weniger Ringe als *optimum* geschossen?“, „Wurden mehr Ringe als *optimum* geschossen?“ sowie „Wurden genau *optimum* Ringe geschossen?“ angearbeitet wurden, wird der Empfehlungstext an den Sportschützen generiert, indem folgende Aktionen durchgeführt werden:

1. Zuerst wird geprüft, ob das Ringziel schon erreicht wurde, nicht mehr erreichbar ist oder der letzte Schuss schon geschossen wurde. Wird diese Frage mit „Ja“ beantwortet, wird die Empfehlung generiert, indem folgende Aktionen ausgeführt werden:

2. Wurde gerade der vorletzte Schuss eingegeben, wird als Empfehlung die Differenz zwischen *gesamt* und der Zielringzahl *ziel* ausgegeben, also die Zahl der noch zu schießenden Ringe.

3. Sonst wird geprüft, ob nicht gerade der letzte Schuss eingegeben wurde, wo eine Schussempfehlung keinen Sinn hätte.

4. Falls ja, wird die Empfehlung für den nächsten Schuss generiert:

5. Zuerst erfolgt die Ausgabe der Information, dass jetzt die Schussempfehlung kommt.

6. Dann wird das Minimum[6] für den nächsten Schuss berechnet und, wenn es größer als 0 ist, ausgegeben.

7. Danach wird (um den Sportschützen emotional zu entlasten) die kleinste Empfehlung gesucht und ausgegeben, wenn sie nicht dem Minimum entspricht.

Wenn das Ringziel schon erreicht wurde, nicht mehr erreichbar ist oder der letzte Schuss schon geschossen wurde, wäre eine Schussempfehlung nicht mehr berechenbar und es wird deshalb nicht versucht, sie zu berechnen.
Eine detaillierte Beschreibung der Umsetzung des Hauptalgorithmus sowie eine Übersicht über alle Variablen, die in diesem verwendet werden, befindet sich im Anhang.
Auf Wunsch ist gern eine ausführbare Version des Programms „Chancenrechner“ zu Testen der so gegebenen Empfehlungen verfügbar.

[6] siehe Einleitung

2.3 Umsetzung in Java

2.3.1 Zur Umsetzung der grafischen Benutzeroberfläche (GUI) und der Ein- und Ausgabe

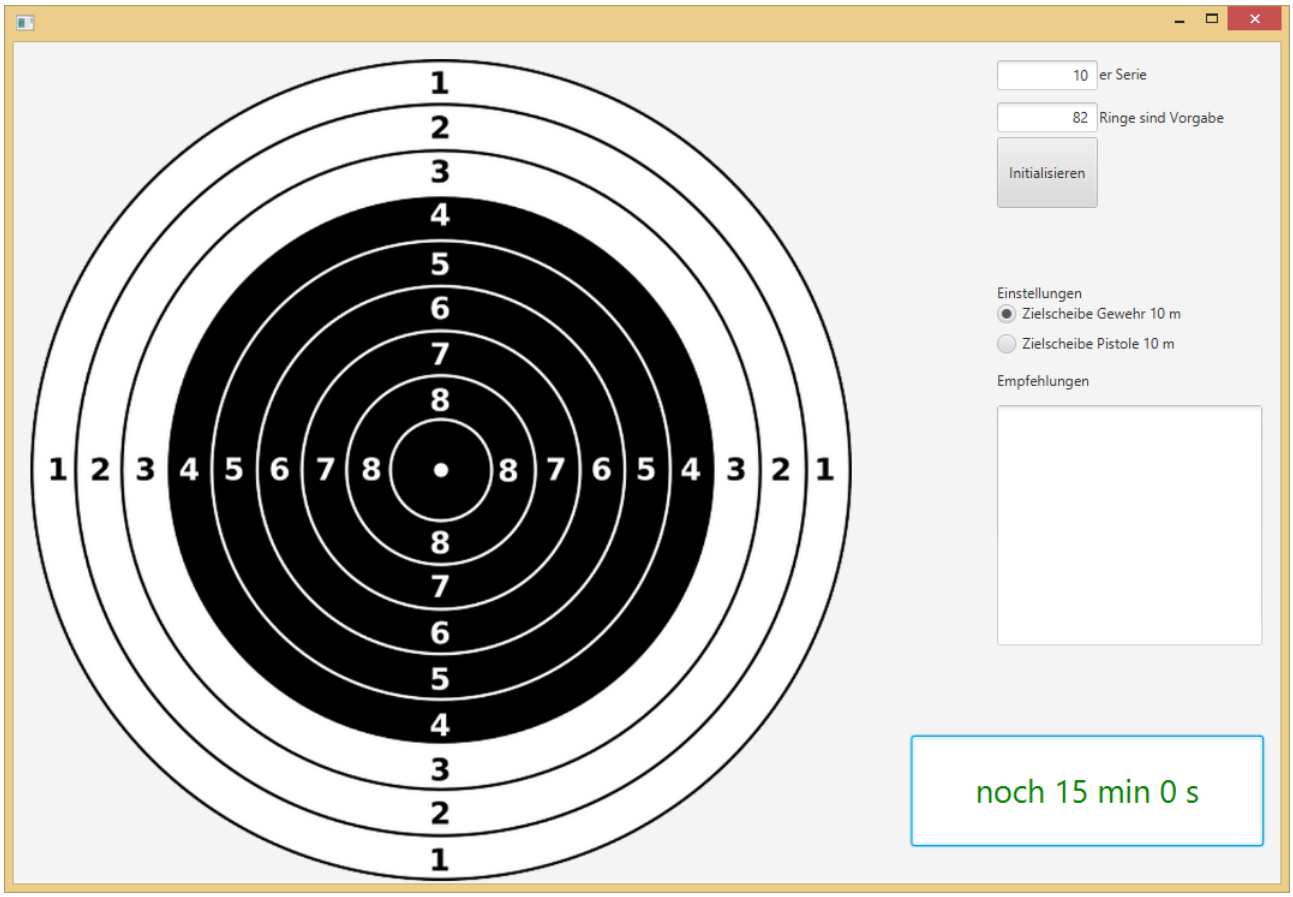

Abbildung 2: Die grafische Benutzeroberfläche des Programms „Chancenrechner“

Die Benutzeroberfläche ist grundlegend in zwei Bereiche aufgeteilt: die Zielscheibe zur Eingabe der einzelnen Schüsse und Ausgabe des Protokolls links und das Ein/Ausgabesegment rechts. Dieses enthält zwei Textfelder, in die Schusszahl und Zielringzahl eingegeben werden. Darunter kann mit zwei RadioButtons die anzuzeigende Zielscheibe ausgewählt werden, aktuell werden „Luftpistole 10m“ und „Luftgewehr 10m“ unterstützt.
Zudem gibt es weiter unten eine Textausgabefläche, in welche die Schussempfehlungen ausgegeben werden. Ganz unten ist der Timer integriert. Es handelt sich hierbei um ein Textfeld mit großer Schriftgröße, um immer gut lesbar zu sein. Solange noch Zeit übrig ist, wird diese in grüner Schriftfarbe dargestellt, ist sie abgelaufen, steht im Timer in roter Schrift „Zeit abgelaufen!“.
Zuletzt gibt es etwa auf mittlerer Höhe im rechten Segment noch einen Mehrfunktionsbutton. Zu Programmstart und immer nach Abschluss einer Serie zeigt er „Initialisieren“. Klickt man ihn an, wird man aufgefordert, seinen Namen für das Protokoll einzugeben, die Parameter für Schuss- und Zielringzahl werden auf Eingabefehler geprüft und ggf. ein Dialog zur Korrektur dieser angezeigt. Des Weiteren werden alle Markierungen auf der Zielscheibe gelöscht, die Empfehlung ausgegeben, möglichst bei jedem Schuss den insgesamt nötigen Durchschnitt zu

schießen und der Timer auf die insgesamt für diese Schussserie vorgegebene oder berechnete Zeit eingestellt. Zum Schluss ändert der Button seinen Titel in „Start“.
Zeigt der Button „Start“, startet er beim Klick auf ihn den Timer, ermöglicht es, auf der Zielscheibe Schüsse einzugeben und verhindert das Wechseln der Zielscheibe. Sein Titel wird auf „Stopp“ gesetzt.
Zeigt der Button „Stopp“, bricht er bei einem Klick die aktuelle Schussserie ab, stoppt den Timer, ermöglicht das Wechseln der Zielscheibe und ändert seinen Titel in „Initialisieren“.
Läuft eine Schussserie, kann durch Klick auf die Zielscheibe an der Stelle des Treffers dessen Schusswert eingegeben werden; das Programm bietet die Möglichkeit, bei Fehlern diesen manuell zu korrigieren. Ein Klick startet jeweils die Empfehlungsberechnung für den nächsten Schuss. Nach der erfolgreichen Eingabe bleibt eine Markierung mit Ort des Schusses und dessen Nummer zurück.
Nach Abschluss der Serie wird auf die Zielscheibe Name und Uhrzeit sowie die Auswertung mit Schusszahl, vorgegebenen Ringen (dem Ringziel), erreichten Ringen, Durchschnitt, Erreichen der Vorgabe und Zeitbedarf geschrieben.
Die Zielscheibe mit Schüssen und Bewertung wird am Ende als PNG-Bild auf dem Desktop gespeichert, dazu ein schriftliches Protokoll mit allen Programmein- und ausgaben der aktuellen Serie. Zudem wird alle 10 Schuss die Scheibe gespeichert und geleert, um sie nicht zu sehr zu füllen (Übersichtlichkeit). Eine mögliche Erweiterung wäre es, Bild- und Textprotokoll zusammen als PDF zu speichern.
Die technische Umsetzung der grafischen Benutzeroberfläche erfolgt mittels einer auf FXML[7] basierenden Java- Benutzeroberfläche. Auf einem Pane[8] liegt ein ImageView[9], darauf wird ein Canvas[10] angeheftet (als Zeichenfläche für die Kreise um die eingegebenen Schüsse), an das Pane ist ein MouseListener[11] angeheftet.

2.3.2 Umsetzung wichtiger Unteralgorithmen

Umsetzung der Eingabe eines Schusswertes

Das Bild der Zielscheibe liegt als PNG vor und ist wie unter 2.3.1 beschrieben in die grafische Benutzeroberfläche integriert. Der Algorithmus zur Ermittlung des Schusswertes wird bei einem Klick auf dieses Bild gestartet, wenn eine Serie läuft. Dabei liefert der angeheftete MouseListener die Koordinaten des Klicks[12]. Die Koordinaten der Mitte der Zielscheibe sind bekannt. Somit kann der Abstand *a* eines Schusses von der Zielscheibenmitte ermittelt werden durch: $\Delta x = |x1 - x|$ und $\Delta y = |y1 - y|$ durch $a = \sqrt{\left((x1 - x)^2 + (y1 - y)^2\right)}$ nach Satz des Pythagoras.

[7] Eine Auszeichnungssprache, deren Aufgabe es ist, grafische Benutzeroberflächen zu erstellen. Diese werden später in ein Java-Programm hineingeladen und dort verwendet.

[8] Ein Oberflächenelement, auf dem andere Elemente angeheftet werden können.

[9] Ein Oberflächenelement, welches Bilder anzeigen kann.

[10] „Canvas“ (engl.) : „Leinwand“; ein Oberflächenelement, auf dem durch das Programm gezeichnet werden kann.

[11] Ein Objekt, das bei einem Klick auf das Oberflächenelement, an das es angeheftet ist, eine im Programm festgelegte Aktion ausführt, hier eine Schussempfehlung für den eingegebenen Schuss berechnet.

[12] Der Koordinatenursprung liegt in der linken oberen Ecke des quadratischen Panes, dessen Inkreis die Zielscheibe ist.

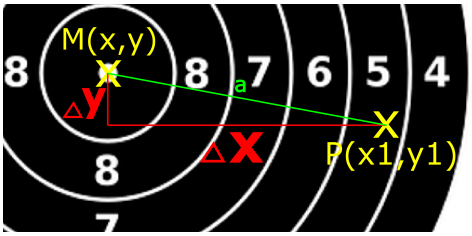

Abbildung 3: Illustration der Herleitung der Berechnung des Abstandes eines Schusses von der Mitte

Durch den Vergleich von *a* mit den Abständen der jeweiligen Ringe von der Mitte kann nun dem Schuss eine Ringzahl zugeordnet werden.
Diese Methode hat den Nachteil, dass die Eingabe besonders an den Rändern der Ringe nicht 100% genau ist und die Zielscheibe immer in einer Auflösung von 800 * 800 Bildpunkten dargestellt werden muss, da die gespeicherten Abstände Absolutwerte sind. Die Darstellung ist für Laptops mit Monitoren mit kleineren Bildschirmen optimiert, jedoch funktioniert sie auch bei höheren Auflösungen.
Für den Fall, dass die Eingabe einmal nicht funktionieren sollte, schafft die Möglichkeit, sie in einem JOptionPane[13] zu korrigieren, Abhilfe.

Umsetzung der Protokollerstellung

Für diese Methode wird zunächst eine neue Instanz der Klasse Calendar erstellt. Diese Instanz enthält das aktuelle Datum und die aktuelle Uhrzeit.
Der Dateiname des Protokolls besteht aus dieser Uhrzeit, die in einem String[14] namens *Name* hinter dem Wort „Protokoll“ gespeichert wird.
Die statistische Auswertung der aktuellen Schussserie wird nun mittels einer Instanz der Klasse Graphics auf die Zielscheibe geschrieben. Sie besteht aus Name und Uhrzeit sowie Schusszahl, Zielringzahl, erreichte Ringe, Durchschnitt, Erreichen der Vorgabe und Zeitbedarf.
Das Pane *flaeche*, auf dem das ImageView, welches die Zielscheibe enthält, sowie die Zeichenfläche mit allen Markierungen und der Statistik, liegen, stellt die Methode „snapshot“ bereit. Diese liefert ein Bildschirmfoto der Zielscheibe mit den Markierungen und der Statistik auf ihr. Dieses Foto wird in einem WritableImage[15] zwischengespeichert und mittels ImageIO[16] auf den Desktop geschrieben.
Für das schriftliche Protokoll wurden alle Programmein- und Ausgaben mit Ausnahme der Zeit im String *Protokoll* gespeichert. Dieser wird nun nur noch in eine Textdatei mit dem selben Dateinamen ebenfalls auf dem Desktop gespeichert.

[13] Eine Klasse, die einfache Fenster mit z.B. Warnmeldungen oder Eingabemöglichkeiten anzeigen kann.
[14] Eine Klasse, deren Instanzen Zeichenketten (Sätze) speichern können.
[15] Eine Klasse, deren Instanzen (Objekte) Bilder speichern können.
[16] Eine Klasse, die Bilder lesen und schreiben kann.

Umsetzen der Zeitfunktion

Immer beim Initialisieren des Programms wird die Gesamtzeit ermittelt. Es gibt hierbei feste Zuordnungen (z.B. 10 Schuss zu 15 Minuten), standardmäßig rechnet das Programm mit einer Minute für einen Schuss. Diese Gesamtzeit wird einmal als *gesamtzeit* und einmal als *zeit* in einer Variable mit diesen Namen gespeichert.
In einer Instanz namens *TimerTask* der Klasse Runnable[17] ist die Aktion gespeichert, die jede Sekunde ausgeführt werden soll:

1. Zuerst wird geprüft, ob *zeit* noch größer ist als 0.
2. Falls ja, wird *zeit* um 1 Sekunde gesenkt und in grüner Farbe im Textfeld *Zeitanzeiger* in Minuten und Sekunden getrennt ausgegeben.
3. Sonst wird im Textfeld in roter Farbe „Zeit abgelaufen!“ angezeigt.

Die Instanz der Klasse ScheduledExecutorService[18] *exec* führt TimerTask jede Sekunde aus. Diese Ausführung ist der ScheduledFuture-Liste[19] *result* zugewiesen. Die Ausführung von TimerTask würde endlos laufen, auch wenn die aktuelle Serie schon vorbei ist, aber *result* kann die Ausführung beenden.

2.4 Zeitlicher Ablauf / Projektentwicklung

Dieses Projekt wurde in regelmäßigen Treffen mit einem Trainer des Sportschützenvereins „Bürgeler-SG e.V.“ und einem Informatiklehrer erarbeitet.
Im Juli 2015 wurde beim ersten Treffen eine grundlegende Konzeption des Programms besprochen.
Während der folgenden Sommerferien wurde ein erster Prototyp des Algorithmus entwickelt. Es handelte sich anfangs noch um ein in der Programmiersprache C++ geschriebenes Programm, welches ohne Visualisierung durch eine grafische Benutzeroberfläche (GUI) in einem DOS-Fenster lief. Diese ursprüngliche Version ermittelte Empfehlungen nach der jeweiligen Abweichung vom Optimum (x Ringe zu wenig getroffen bedeutet Optimum + x Ringe schießen usw.). Wenn die dadurch berechnete Empfehlung nicht mehr praktisch realisierbar (kleiner 0 oder größer 10) war, wurde diese Zahl schrittweise automatisch reduziert und die Anzahl der Schüsse auf diese Zahl zum Ausgleich schrittweise erhöht. Die so berechneten Empfehlungen verließen den praktisch sinnvollen Rahmen und zielten auf viel zu hohe Gesamtringzahlen. Zudem wurde das gesamte Array mit den Empfehlungen ausgegeben statt einer einzelnen Empfehlung, außerdem fehlte die Ausgabe des Minimalwertes.
In den ersten Schulwochen des Schuljahres 2015/2016 wurden viele kleine Fehler am Hauptalgorithmus behoben und die Neuberechnung des Optimalwertes eingebaut.

[17] Eine Klasse, deren Instanzen Programmcode enthalten, welcher vom Nutzer frei bestimmt und beliebig ausgeführt werden kann.
[18] Eine Klasse, deren Objekte ausführbare Aktionen, z.B. der Instanzen der Klasse Runnable, zeitlich periodisch wiederholen können.
[19] Hier eine Liste aller einzelner Aufrufe von *TimerTask*; welche den Zugriff auf diese erlaubt, um z.B. die Ausführung von *TimerTask* zu beenden.

Beim zweiten Treffen wurde der Prototyp dem Trainer zum Testen übergeben sowie einige Details verbessert.
Beim dritten Treffen wurden Verbesserungen an der Ausgabe besprochen[20] und der Algorithmus so erweitert, dass er bis zum Ende der Schussserie durchläuft (auch wenn das Ringziel nicht mehr erreichbar ist).
Danach wurde die Ausgabe des Minimalwertes hinzugefügt und die Empfehlungsausgabe so geändert, dass immer nur eine Empfehlung ausgegeben wird. Zu diesem Zeitpunkt nahm die Idee einer grafischen Visualisierung Gestalt an. Es wurde ein zweiter Prototyp von „Chancenrechner“ als Java-Programm mit grafischer Benutzeroberfläche entwickelt, der zu diesem Zeitpunkt nur Ein- und Ausgabe beherrschte.
Beim vierten Treffen wurden beide Prototypen übergeben sowie an beiden einige Details verbessert, besonders die Hinzufügung der Zeitbetrachtung.
Danach wurde der Hauptalgorithmus in den Prototyp mit grafischer Benutzeroberfläche implementiert und die grafische Benutzeroberfläche bis zum jetzigen Stand weiterentwickelt.
Beim fünften Treffen wurde aus praktischen Erwägungen heraus entschieden, ausschließlich die Entwicklung des Prototypen mit grafischer Benutzeroberfläche weiter zu verfolgen. Es erfolgten Detailverbesserungen an der grafischen Benutzeroberfläche und dem Algorithmus, beispielsweise, dass immer nach 10 Schüssen die Scheibe als Bild gespeichert und anschließend automatisch geleert wird (Grund: die Scheibe wurde mit mehr als 10 Treffern unübersichtlich, außerdem wird die reale Scheibe im Wettkampf auch nach 10 Schüssen erneuert).
Beim sechsten Treffen erfolgten noch Detailverbesserungen an der Zeitfunktion und der Oberfläche allgemein.
Während der gesamten Arbeitszeit wurden permanent Fehler im Programm gesucht und ausgebessert, wobei auch die Testprotokolle herangezogen wurden.

[20] Siehe die angehängten Protokolle vor und nach diesem Treffen

3 Bewertung des Programms

3.1 Fehlerdiskussion

In einer frühen Version des C++- Programms trat der Fehler auf, dass durch einen Umwandlungsfehler von Ganz- in Gleitkommazahlen die Berechnung des Minimums versagte. Auch der Hauptalgorithmus selbst neigte am Anfang zu Fehlern, die jedoch behoben wurden. Aufgrund der geringeren Anfälligkeit gegenüber Programmfehlern und der besseren Möglichkeiten einer grafischen Visualisierungen wechselten wir die verwendete Programmiersprache zu Java.
Ein Problem trat auch unter Java (bei einem leistungsmäßig sehr schwachen Laptop) auf: Bei der Korrektur eines Eingabefehlers kam es zu einem Absturz des Programms, wenn man das entsprechende JOptionPane zu lange angezeigt ließ. Ein Lösungsversuch schlug fehl. Dieses Problem konnte bisher nicht gelöst werden, tritt aber bei adäquater Hardware nicht auf. Deshalb verfolgten wir es nicht weiter. Den zukünftigen Nutzern werden wir allerdings, wie bei handelsüblichen Programmen üblich, die Hardware-Mindestanforderungen mitteilen.
Teilweise kommt es noch zu Verzögerungen beim Protokollieren, da die relativ großen Datenmengen der Bildprotokolle erst geschrieben werden müssen. Eine denkbare Lösung wäre Multithreading (Aufwand-Nutzen-Verhältnis ungünstig) oder eine schnellere Festplatte (SSD). Diese Verzögerungen liegen aber lediglich im Bereich von 1-2 Sekunden, sind also gerade so spürbar und haben sich im Alltag als unproblematisch erwiesen.
Das finale Programm läuft nach bisherigem Erkenntnisstand stabil, im Rahmen der praktischen Erprobung sind keine Abstürze mehr aufgetreten.

3.2 Vergleich des Programms mit der Zielsetzung

Das Programm gibt einem Sportschützen (lt. Hr. Kunze) sinnvolle und hilfreiche Informationen und hilft ihm, über die aktuelle Serie den Überblick zu bewahren und die aktuelle Leistung einzuschätzen (letzter Schuss, der allgemeine Schusstrend mit den Markierungen der Schüsse auf der Scheibe. . .).
Besonders die Timerfunktion des Programms wird als echte Hilfe gesehen.
Das Programm ist aktuell nutzbar für das Training mit Luftgewehr und Pistole auf 10 m Entfernung durch die eingebauten sportordnungsgerechten Scheiben, andere Scheiben können jedoch leicht eingebaut werden. Jede mögliche Kombination von Schüssen und Zielringzahl kann simuliert werden. Sämtliche Zeitvorgaben im Programm sowie die angezeigten Zielscheiben basieren auf der deutschen Sportordnung.
Die Protokolle sind für den Trainer interessant, besonders der Durchschnitt. Sie können archiviert und später erneut zur Auswertung des Trainingserfolges der Sportschützen herangezogen werden.
Die Ein-und Ausgabe ist relativ genau, die Bedienung einfach und eingabefehlerresistent gelungen.
Die Sportschützen, die dieses Programm getestet haben, und deren Trainer mögen es und bezeichnen es als echte Hilfe beim Training.

3.3 Ausblick

Dieses Projekt kann noch erweitert werden. Denkbar ist zum Beispiel eine Vertiefung der Statistikfunktionen inklusive der Speicherung von Bild- und Textprotokoll in einer PDF-Datei. Da die Smartphone-Entwicklung zu immer größeren Displays tendiert und damit eine Darstellung des Programms auf Mobilgeräten sinnvoll wird, ist auch die Ergänzung des Programms durch eine Android-App für den Sportschützen geplant. Diese könnte ihm vor Ort die Empfehlungen anzeigen und sich zum Beispiel auch mit dem PC-Programm des Trainers synchronisieren. Zuletzt ist noch geplant, den reinen Hauptalgorithmus in ein Programm zu importieren, welches an eine elektronische Schießscheibe angeschlossen ist und von dieser die aktuellen Schusswerte bekommt.

4 Literaturverzeichnis

Bildquellen

- https://pixabay.com/static/uploads/photo/2012/04/24/16/43/targets-40383_640.png : 01.01.2016 14:10, Bild der Zielscheibe Luftgewehr 10 m im Programm
- http://www.schuetzenbedarf-baur.com/images/Krueger%20Pistole%2025-50%20m%20C50%203101_100.jpg : 01.01.2016 14:11, Bild der Zielscheibe Luftpistole 10 m im Programm
- andere Bilder in der Projektarbeit: selbst erstellt

Hilfe bei der Programmentwicklung

- http://code.makery.ch/blog/javafx-2-snapshot-as-png-image/ : 01.01.2016 14:14, Marco Jacob, JavaFX 2 Snapshot as PNG Image
- http://stackoverflow.com/questions/16128423/how-to-update-the-label-box-every-2-seconds-in-java-fx : 01.01.2016 14:18, Stackoverflow.com, How to update the label box every 2 seconds in java fx?
- http://tomasmikula.github.io/blog/2014/06/04/timers-in-javafx-and-reactfx.html : 01.01.2016 14:19, Tomas Mikula, Timers in JavaFX and ReactFX
- http://stackoverflow.com/questions/16764549/timers-and-javafx : 01.01.2016 14:20, Stackoverflow.com, Timers and javafx
- https://de.wikibooks.org/wiki/Java_Standard:_Datum : 01.01.2016 14:21, de.wikibooks.org, Java Standard: Datum

persönliche Unterstützung

- Thiele, Otto, Informatiklehrer, Carl-Zeiss-Gymnasium, Jena, Art der Unterstützung: Hilfe bei der Themenwahl und Konzeption des Programms, Unterstützung bei der Umsetzung
- Kunze, Bert, Trainer, Bürgeler-SG e.V., Jena, Art der Unterstützung: Hilfe bei der Konzeption des Programms, Test der Programmprototypen im Sportschützenverein, Hilfe bei der Formulierung der Einleitung

5 Anhang

5.1 Übersicht über die Variablen im Hauptalgorithmus und ihre Aufgaben

Variable	Typ	Aufgabe
erstesmal	boolean	speichert, ob gerade der erste Schuss der aktuellen Serie eingegeben wurde
zahl	Array von 11 int-Zahlen	speichert die Empfehlungen, die das Programm geben wird; dabei sind die Indizes die Schusswerte von 0 bis 10 und deren Werte jeweils die Anzahl, wie oft der Wert zu schießen ist
optimum	int	speichert den nötigen Durchschnittswert, um die Zielringzahl zu erreichen; immer aufgerundet bei der Berechnung, um nie zu wenig Ringe zu empfehlen
frischgenullt	boolean	speichert, ob gerade optimum neu berechnet wurde
gesamt	int	speichert die insgesamt geschossene Ringzahl
aktuell	int	speichert den aktuell eingegebenen Schuss
ergebnisse	Array von int-Zahlen; für jeden Schuss der Serie eine	speichert die einzelnen Ringzahlen der Schüsse
letzter	int	speichert die Ringzahl des letzten Schusses
echteschuesse	int	Nummer des aktuellen Schusses
ziel	int	speichert die Zielringzahl
nichtmehrzuerreichen	boolean	speichert, ob die Zielringzahl nicht mehr zu erreichen ist oder bereits erreicht wurde
i	int	speichert die Nummer des aktuellen Durchlaufs
nurabgearbeitet	int	speichert, ob eine Empfehlung eingehalten wurde
Guthaben	int	speichert die Differenz zwischen der Zielringzahl und dem Wert, den man erhält, wenn man die Empfehlungen befolgt
zwischensumme	int	Mehrzweck-Variable: speichert die Zahl aller verbleibender Schüsse
j	int	speichert die aktuelle Empfehlung

Tabelle 1: Übersicht über die Variablen im Hauptalgorithmus und ihre Aufgaben

5.2 Ausführliche Dokumentation des Hauptalgorithmus

Im Folgenden wird der Hauptalgorithmus des Programms in Struktogrammen dargestellt und schrittweise erklärt.
Für Leser, die sich für Details der Umsetzung des Hauptalgorithmus im Programm interessieren, steht auf Nachfrage der kommentierte Sourcecode von „Chancenrechner“ zur Verfügung.
Verschiedene Arten des auszuführenden Codes wurden farblich hervorgehoben:

- Lachsrot markiert sind für den Programmablauf nötige Festlegungen, zum Beispiel Variableninitialisierungen.
- Gelb markiert sind Ausgaben des Algorithmus’.
- Hellgrün markiert sind die Teile des Hauptalgorithmus’, die ausgeführt werden, wenn weniger Ringe als im Durchschnitt nötig geschossen wurden.
- Türkisblau markiert sind die Teile des Hauptalgorithmus’, die ausgeführt werden, wenn mehr Ringe als im Durchschnitt nötig geschossen wurden.
- Pink markiert sind die Teile des Hauptalgorithmus’, die standardmäßig ausgeführt werden müssen.

5.2.1 Algorithmus: Variableninitialisierungen und nötige Festlegungen

Anmerkung: eine Übersicht über alle Variablen in der Reihenfolge ihres Auftretens sowie deren Funktion befindet sich im Anhang.

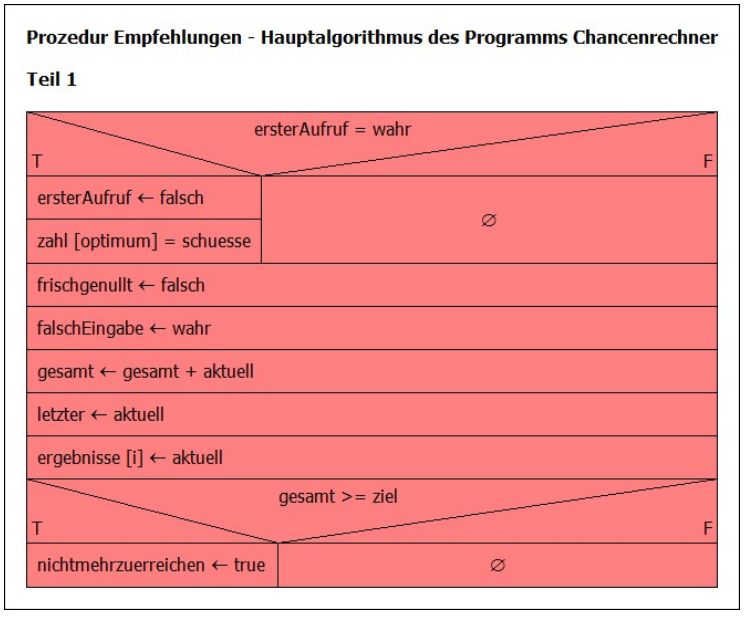

Abbildung 4: Teil 1 des Hauptalgorithmus als Struktogramm

Der Hauptalgorithmus wird bei jedem Schuss mit dessen aktuellen Wert neu aufgerufen.
Das Array von 11 Integer-Zahlen (Ganzzahlen von 0 bis 10) *zahl* speichert alle Empfehlungen, wobei der jeweilige Index die Schussempfehlung und der Wert im Array dessen Anzahl ist.

Wenn also $zahl[5] = 2$ empfiehlt das Programm, mit den restlichen Schüssen zweimal 5 Ringe zu treffen.
Beim ersten Aufruf des Algorithmus in der aktuellen Schussserie muss das Array so initialisiert werden, dass es empfiehlt, bei jedem Schuss den Optimalwert zu schießen. Also wird $zahl[optimum] = schuesse$ gesetzt.
Der Wahrheitswert *frischgenullt* muss auf falsch gesetzt werden, weil das Optimum in diesem Durchlauf noch nicht neu berechnet wurde.
Der Wahrheitswert *falscheingabe* ist algorithmisch nicht bedeutend, hier aber erwähnt.
Bei jedem Schuss muss der Gesamtringzahl *gesamt* noch der Wert des aktuellen Schusses hinzuaddiert werden. *gesamt* wird vor jeder Schussserie auf 0 gesetzt.
letzter wird bei jedem Durchlauf auf den Wert des zuletzt eingegebenen Schusses, *aktuell*, gesetzt.
Das Array von Integer-Zahlen *ergebnisse* speichert die einzelnen Schussergebnisse, um später statistische Erweiterungen wie z.B. eine Trendanalyse hinzufügen zu können. Jeder neu eingegebene Schuss wird dazu in der Integer-Zahl mit dem Index seiner Nummer minus 1 gespeichert (Java-Indizes zählen von 0 bis Anzahl-1).
Der Algorithmus wird nur ausgeführt, wenn man seine Zielringzahl noch nicht erreicht hat, also wird dies vor jedem Durchlauf erst einmal geprüft.

5.2.2 Algorithmus: grundlegende Arbeitsweise

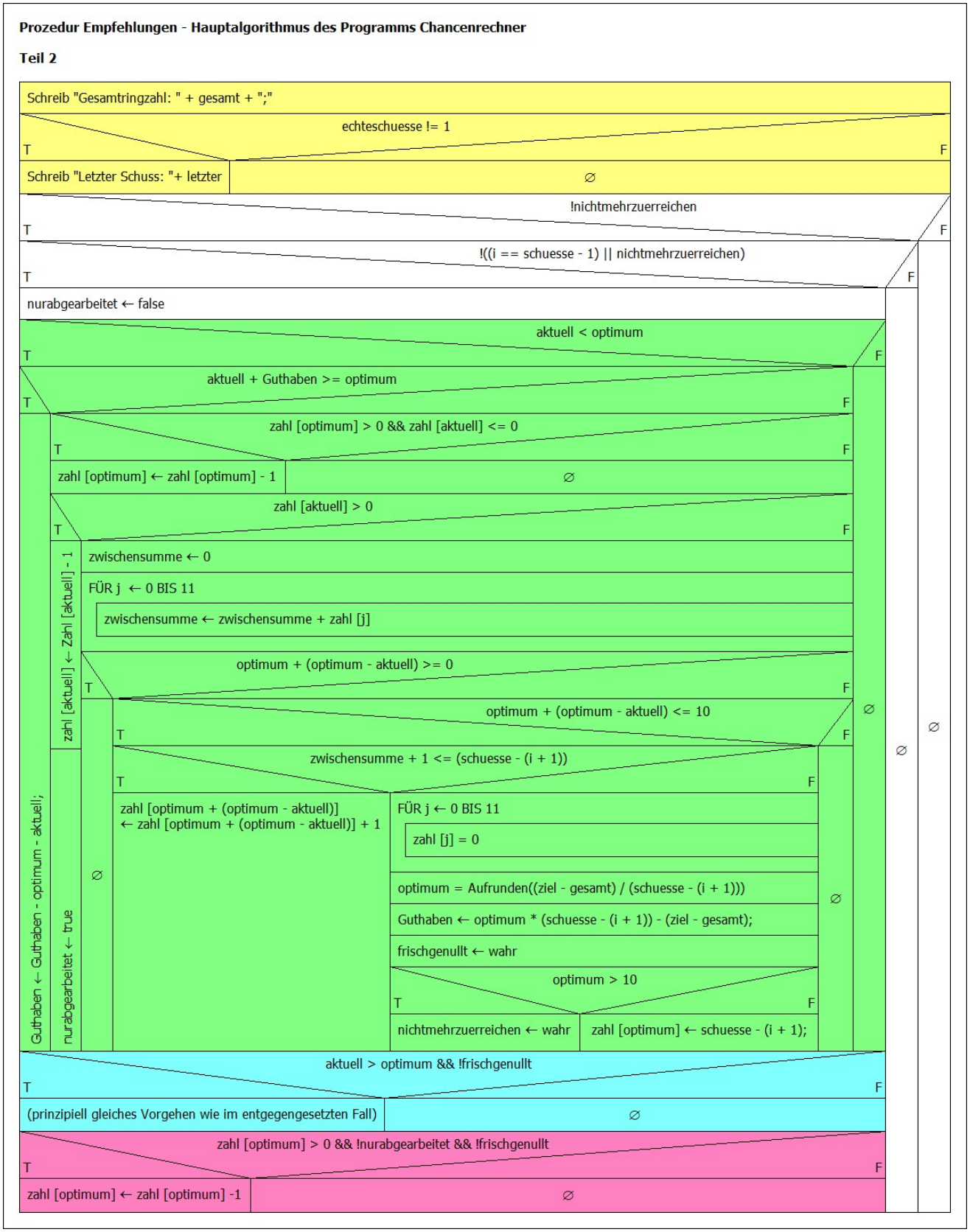

Abbildung 5: Teil 2 des Hauptalgorithmus als Struktogramm

Bei jedem Schuss werden dem Sportschützen zur besseren Orientierung zu seinem Ziel hin seine Gesamtringzahl und sein letzter Schuss ausgegeben.
Für alle Empfehlungsberechnungen gilt: Diese werden nur ausgeführt, wenn das Ziel noch zu erreichen ist und nicht bereits erreicht wurde (Wahrheitswert *nichtmehrzuerreichen*), was nur Programmabstürze oder unausführbare Empfehlungen (z.B. 12 Ringe zu schießen) zur Folge hätte, und es sich nicht um den letzten Schuss handelt, wo Empfehlungen nicht mehr sinnvoll wären.
Wird dies erfüllt, wird der Wahrheitswert *nurabgearbeitet*, welcher beschreibt, dass eine empfohlene Ringzahl geschossen wurde, auf *falsch* gesetzt, da dies später noch überprüft werden muss.
Danach wird geprüft, ob weniger Ringe geschossen wurden, als *optimum* vorgibt. Nehmen wir an, dies sei gegeben, wird das Kernstück des Hauptalgorithmus ausgeführt:

1. Die Variable *Guthaben* speichert die Differenz der Ringe, die man erhält, wenn man den vorgeschlagenen Durchschnitt erreicht, und der Zielringzahl. Ist *Guthaben* größer oder gleich der aktuellen Abweichung, wird *Guthaben* um diese verringert. Damit wird realisiert, dass die Abweichung ausgeglichen und der Schuss wie ein Schuss, der *optimum* Ringe getroffen hat, behandelt wird.

2. Ist *Guthaben* kleiner als die Abweichung von *optimum*, dann wird geprüft, ob die aktuelle Ringzahl eine Schussempfehlung war. Falls ja, wird die Ganzzahl mit dem Index der aktuellen Ringzahl im Array *zahl* um 1 erniedrigt und im Wahrheitswert *nurabgearbeitet* gespeichert, dass eine Empfehlung befolgt wurde.

3. Falls nein, wird geprüft, ob noch empfohlen wird, *optimum* Ringe zu schießen. Ist dies gegeben, wird *zahl[optimum]* um 1 erniedrigt, da *optimum* in der im Folgenden generierten Empfehlung enthalten ist.

4. Danach wird die Abweichung des aktuellen Wertes von *optimum* ermittelt und geprüft, ob *optimum* plus diese Empfehlung im Bereich zwischen 0 und 10 Ringen liegt. Falls ja, wird empfohlen, *optimum* plus diese Abweichung Ringe zu schießen, damit die Summe des aktuellen und des nächsten Schusses zweimal *optimum* beträgt.

5. Fällt die zum Ausgleich nötige Zahl aus diesem Intervall, muss *optimum* neu berechnet werden. Dafür werden alle Empfehlungen auf 0 gesetzt und *optimum* als Quotient der noch zu erreichenden Ringzahl und der verbleibenden Schusszahl neu berechnet. Dabei muss *optimum* aufgerundet werden, um nicht zu zu wenigen Ringen als Endergebnis zu führen.

6. *Guthaben* wird als Differenz $übrigeSchüsse * optimum + gesamt - ziel$ neu berechnet.

7. Im Wahrheitswert *frischgenullt* wird gespeichert, dass gerade ein neues *optimum* berechnet wurde.

8. Zuletzt wird geprüft, ob *optimum* größer als 10 ist. Falls ja, wird im Wahrheitswert *nichtmehrzuerreichen* gespeichert, dass man die Zielringzahl nicht mehr erreichen kann, sonst wird im Array *zahl* die Empfehlung gespeichert, bei jedem übrigen Schuss *optimum* Ringe zu schießen.

Im möglichen Fall, dass mehr Ringe als *optimum* geschossen wurden, ist das Vorgehen adäquat.
Zuletzt wird noch geprüft, ob die Anzahl der Empfehlung, *optimum* zu schießen (*zahl[optimum]*) größer 0, *optimum* nicht neu berechnet und nicht nur die Ringzahl einer Empfehlung geschossen wurde. Ist dies gegeben, wird *zahl[optimum]* noch um 1 verringert, da entweder *optimum* Ringe geschossen wurden oder *optimum* in der generierten Empfehlung enthalten ist.

5.2.3 Algorithmus: Generierung der Empfehlungsausgabe

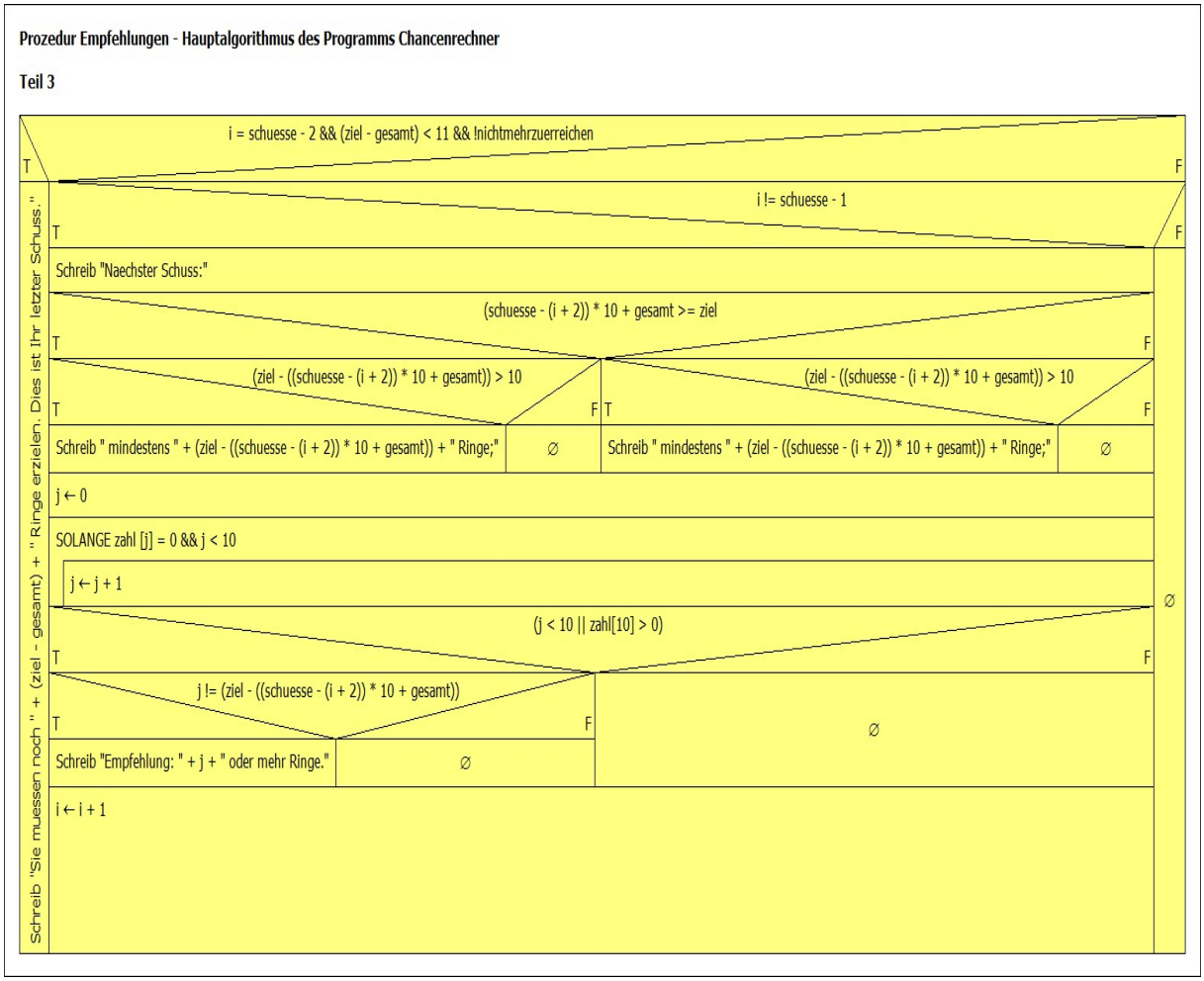

Abbildung 6: Teil 3 des Hauptalgorithmus als Struktogramm

Wurde gerade der vorletzte Schuss eingegeben, wird als Empfehlung die Differenz zwischen *gesamt* und der Zielringzahl *ziel* ausgegeben, also die Zahl der noch zu schießenden Ringe. Sonst wird geprüft, ob nicht gerade der letzte Schuss eingegeben wurde, wo eine Schussempfehlung keinen Sinn hätte.
Falls ja, wird die Empfehlung für den nächsten Schuss generiert:

1. Zuerst erfolgt die Ausgabe der Information, dass jetzt die Schussempfehlung kommt.
2. Dann wird das Minimum[21] berechnet und, wenn es größer als 0 ist, ausgegeben.
3. Danach wird (um den Sportschützen emotional zu entlasten) die kleinste Empfehlung gesucht und ausgegeben, wenn sie nicht dem Minimum entspricht.

Zuletzt wird die Zahl der Durchläufe durch den Hauptalgorithmus *i* erhöht.

[21] siehe Abschnitt 2.2.1

5.3 Testprotokolle

5.3.1 Testprotokoll vor dem dritten Treffen

Programm Chancenrechner Version 0.4 ALPHA
Entwickler: Eric Ackermann- Carl-Zeiss-Gymnasium Jena
Experimentelle Programmversion, Nutzung auf eigene Gefahr.
Schuetze: 10
Schuesse: 5
Zielpunkte: 32
Empfehlungen werden gegeben zum Erreichen von 35 Punkten.
Ihr Ziel: durschschnittlich 7 Punkte.
Schuss 1: 7
Ergebnisse der Schuesse: 7;
Bisherige Gesamtpunktzahl: 7
Sie muessen noch 4 mal 7 Treffer erzielen.
Schuss 2: 7
Ergebnisse der Schuesse: 7;7;
Bisherige Gesamtpunktzahl: 14
Sie muessen noch 3 mal 7 Treffer erzielen.
Schuss 3: 7
Ergebnisse der Schuesse: 7;7;7;
Bisherige Gesamtpunktzahl: 21
Sie muessen noch 2 mal 7 Treffer erzielen.
Schuss 4: 7
Ergebnisse der Schuesse: 7;7;7;7;
Bisherige Gesamtpunktzahl: 28
Sie muessen noch 4 Treffer erzielen. Dies ist Ihr letzter Schuss.
Schuss 5: 7
Ergebnisse der Schuesse: 7;7;7;7;7;
Mit 35 Punkten haben Sie Ihr Ziel von 32 Punkten erreicht. Ihnen stehen noch 0 Schuesse zur Verfuegung.
Sie haben 35 Punkte erzielt.
Sie haben durchschnittlich 7 Treffer erzielt.
Herzlichen Glueckwunsch, Sie haben Ihre Zielpunktzahl erreicht.

5.3.2 Testprotokoll nach dem dritten Treffen

Programm Chancenrechner Version 0.5 ALPHA
Entwickler: Eric Ackermann- Carl-Zeiss-Gymnasium Jena
Experimentelle Programmversion, Nutzung auf eigene Gefahr.
Schuetze: Alex
Serie: 10
Ringe zu erreichen: 78
Das Ziel: durschschnittlich 7.8 Ringe.
Schuss 1: 9
Gesamtringzahl: 9;
Naechster Schuss:
Empfehlung: 8 oder mehr Ringe.
Schuss 2: 10
Gesamtringzahl: 19;
Naechster Schuss:
Empfehlung: 8 oder mehr Ringe.
Schuss 3: 7
Gesamtringzahl: 26;
Naechster Schuss:
Empfehlung: 8 oder mehr Ringe.
Schuss 4: 8
Gesamtringzahl: 34;
Naechster Schuss:
Empfehlung: 8 oder mehr Ringe.
Schuss 5: 7
Gesamtringzahl: 41;
Naechster Schuss:
Empfehlung: 8 oder mehr Ringe.
Schuss 6: 7
Gesamtringzahl: 48;
Naechster Schuss:
Empfehlung: 8 oder mehr Ringe.
Schuss 7: 8
Gesamtringzahl: 56;
Naechster Schuss: mindestens 2 Ringe;
Empfehlung: 8 oder mehr Ringe.
Schuss 8: 8
Gesamtringzahl: 64;
Naechster Schuss: mindestens 4 Ringe;
Empfehlung: 8 oder mehr Ringe.
Schuss 9: 7
Gesamtringzahl: 71;

Sie muessen noch 7 Ringe erzielen. Dies ist Ihr letzter Schuss.
Schuss 10: 8
Gesamtringzahl: 79;
Sie haben 79 Ringe erzielt und dafuer 1 Minuten 25 Sekunden gebraucht.
Sie haben durchschnittlich 7.9 Ringe erzielt.
Herzlichen Glueckwunsch, Sie haben Ihre Zielringzahl erreicht.