

1. Summe der Preise

| Umgangssprache | Aktivitätsdiagramm |
|--|---|
| <p>Setze eine Variable „s“ (für Summe) auf 0.</p> <p>Wiederhole für jedes Element des Arrays:</p> <ul style="list-style-type: none">Addiere den Wert des aktuellen Elements zu s | <pre>graph TD; Start(()) --> Init[Variable s = 0]; Init --> LoopStart{ }; LoopStart --> Add[addiere aktuelles Array-Element zu s]; Add --> LoopEnd{ }; LoopEnd -- "[für jedes Element]" --> LoopStart; LoopEnd --> End((()));</pre> |
| Programmcode | |
| <pre>public double summe() { int i; double s; s = 0.0; for (i=0; i<liste.length; i++) { s = s + liste[i]; } return s; }</pre> | |

2. Mittelwerte der Preise

| Umgangssprache | Aktivitätsdiagramm |
|---|---|
| <p>Setze eine Variable „s“ (für Summe) auf 0.</p> <p>Wiederhole für jedes Element des Arrays:</p> <ul style="list-style-type: none">• Addiere den Wert des aktuellen Elements zu s <p>Teile s durch die Anzahl der Elemente</p> | <pre>graph TD Start(()) --> Init[Variable s = 0] Init --> Dec1{ } Dec1 --> Add[addiere aktuelles Array-Element zu s] Add --> Dec2{ } Dec2 -- "[für jedes Element]" --> Dec1 Dec2 --> Calc[Mittelwert = s / Anzahl Elemente] Calc --> End((()))</pre> |
| <p>Programmcode</p> <pre>public double mittelwert() { int i; double s, m; s = 0.0; for (i=0; i<liste.length; i++) { s = s + liste[i]; } m = s / liste.length; return m; }</pre> | |

3. Niedrigster Preis („Minimum“)

Der Algorithmus für das Maximum funktioniert analog.

| Umgangssprache | Aktivitätsdiagramm |
|---|--|
| <p>Setze eine Variable „min“ auf das erste Element</p> <p>Wiederhole für alle Elemente des Arrays ab dem zweiten:</p> <ul style="list-style-type: none"> Falls das aktuelle Element kleiner ist als min, setze min auf das aktuelle Element | <pre> graph TD Start(()) --> Init[Variable min = erstes Array-Element] Init --> D1{ } D1 --> D2{ } D2 -- "[akt. Elem. < min]" --> SetMin[setze min auf das aktuelle Element] D2 -- "[sonst]" --> Merge{ } SetMin --> Merge Merge --> D3{ } D3 -- "[für jedes Element ab dem 2.]" --> D1 D3 --> End((())) </pre> |
| <p>Programmcode</p> <pre> public double minimum() { int i; double min; min = liste[0]; for (i=1; i<liste.length; i++) { if (liste[i] < min) { min = liste[i]; } } return min; } </pre> | |

4. Anzahl Nullen

| Umgangssprache | Aktivitätsdiagramm |
|---|---|
| <p>Setze eine Variable „anzahl“ auf 0.</p> <p>Wiederhole für jedes Element des Arrays:</p> <ul style="list-style-type: none"> Falls das aktuelle Element gleich 0 ist, erhöhe anzahl um 1. | <pre> graph TD Start(()) --> Init[Variable anzahl = 0] Init --> LoopEntry(()) LoopEntry --> Cond1{ } Cond1 -- "[akt. Elem. = 0]" --> Inc[erhöhe anzahl um 1] Cond1 -- "[sonst]" --> Cond2{ } Inc --> Cond2 Cond2 --> Cond3{ } Cond3 --> Cond4{ } Cond4 --> LoopEntry Cond4 --> End((())) </pre> <p>[für jedes Element]</p> |
| <p>Programmcode</p> <pre> public int anzahlNullen() { int i, anzahl; anzahl = 0; for (i=0; i<liste.length; i++) { if (liste[i] == 0) { anzahl = anzahl + 1; } } return anzahl; } </pre> | |