2.6 Example: Health data

Calculating user's age in days

The section presents an example program that computes various health related data based on a user's age using incremental development. ***Incremental development*** is the process of writing, compiling, and testing a small amount of code, then writing, compiling, and testing a small amount more (an incremental amount), and so on.

The initial program below calculates a user's age in days based on the user's age in years. The assignment statement `userAgeDays = userAgeYears * 365;` assigns userAgeDays with the product of the user's age and 365, which does not take into account leap years.

Figure 2.6.1: Health data: Calculating user's age in days.

```
#include <iostream>
using namespace std;

int main() {
   int userAgeYears;
   int userAgeDays;

   cout << "Enter your age in years: ";
   cin  >> userAgeYears;

   userAgeDays = userAgeYears * 365;

   cout << "You are " << userAgeDays << " days old." << endl;

   return 0;
}
```

```
Enter your age in years: 19
You are 6935 days old.
```

Feedback?

PARTICIPATION ACTIVITY 2.6.1: Calculating user age in days.

1) Which variable is used for the user's age in years?

userAgeYears

Check Show answer

Correct

userAgeYears

The variable name userAgeYears clearly indicates the variable will hold the user's age in years.

Similarly, userAgeDays is for the user's age in days.

2) If the user enters 10, what will userAgeYears be assigned?

10

Check Show answer

Correct

10

userAgeYears is assigned with the user-entered value, or 10.

3) If the user enters 10, what is userAgeDays assigned?

3650

Check Show answer

Correct

3650

userAgeDays is assigned with userAgeYears * 365, which is 10 * 365, or 3650.

Feedback?

Considering leap years and calculating age in minutes

The program below extends the previous program by accounting for leap years when calculating the user's age in days. Since each leap year has one extra day, the statement `userAgeDays = userAgeDays + (userAgeYears / 4)` adds the number of leap years to userAgeDays. Note that the parentheses are not needed but are used to make the statement easier to read.

The program also computes and outputs the user's age in minutes.

Figure 2.6.2: Health data: Calculating user's age in days and minutes.

```
#include <iostream>
using namespace std;

int main() {
   int userAgeYears;
   int userAgeDays;
   int userAgeMinutes;

   cout << "Enter your age in years: ";
   cin  >> userAgeYears;

   userAgeDays = userAgeYears * 365;                // Calculate days without leap years
   userAgeDays = userAgeDays + (userAgeYears / 4); // Add days for leap years

   cout << "You are " << userAgeDays << " days old." << endl;

   userAgeMinutes = userAgeDays * 24 * 60;          // 24 hours/day, 60 minutes/hour
   cout << "You are " << userAgeMinutes << " minutes old." << endl;

   return 0;
}
```

```
Enter your age in years: 19
You are 6939 days old.
You are 9992160 minutes old.
```

Feedback?

PARTICIPATION ACTIVITY 2.6.2: Calculating user age in days. ✓

1) The expression `(userAgeYears / 4)` assumes a leap year occurs every four years?

[✓ True] False

✓ **Correct**

Since the program does not ask what year the user was born, the calculation does not account for the absence of leap years in some years that are multiples of 100.

2) The statement `userAgeDays = userAgeDays + (userAgeYears / 4);` requires parentheses to evaluate correctly.

True [✓ False]

✓ **Correct**

Parentheses are not required because / has higher precedence than +. But, the parentheses make the order of evaluation explicit, which makes the programmer's intention clear.

3) If the user enters 20, what is userAgeDays

✓ **Correct**

after the first assignment statement?

[✓ 7300]

7305

The first assignment statement assigns userAgeDays with 20 * 365 or 7300.

4) If the user enters 20, what is userAgeDays after the second assignment statement?

7300

[✓ 7305]

Correct

The second assignment statement assigns userAgeDays with 7300 + (20 / 4), which is 7300 + 5 or 7305.

Feedback?

Estimating total heartbeats in user's lifetime

The program is incrementally extended again to calculate the approximate number of times the user's heart has beat in his/her lifetime using an average heart rate of 72 beats per minutes.

Figure 2.6.3: Health data: Calculating total heartbeats lifetime.

```
#include <iostream>
using namespace std;

int main() {
   int userAgeYears;
   int userAgeDays;
   int userAgeMinutes;
   int totalHeartbeats;
   int avgBeatsPerMinute = 72;

   cout << "Enter your age in years: ";
   cin  >> userAgeYears;

   userAgeDays = userAgeYears * 365;              // Calculate days without leap years
   userAgeDays = userAgeDays + (userAgeYears / 4); // Add days for leap years

   cout << "You are " << userAgeDays << " days old." << endl;

   userAgeMinutes = userAgeDays * 24 * 60;         // 24 hours/day, 60 minutes/hour
   cout << "You are " << userAgeMinutes << " minutes old." << endl;

   totalHeartbeats = userAgeMinutes * avgBeatsPerMinute;
   cout << "Your heart has beat " << totalHeartbeats << " times." << endl;

   return 0;
}
```

```
Enter your age in years: 19
You are 6939 days old.
You are 9992160 minutes old.
Your heart has beat 719435520 times.
```

Feedback?

PARTICIPATION ACTIVITY 2.6.3: Calculating user's heartbeats.

1) Which variable is initialized when declared?

- userAgeYears
- totalHeartbeats
- [✓ avgBeatsPerMinute]

Correct

The variable declaration for avgBeatsPerMinute initializes the variable with 72.

2) If the user enters 10, what value is held in totalHeartbeats after the statement `userAgeDays = userAgeYears * 365;`

- 3650
- 5258880
- [✓ Unknown]

Correct

totalHeartbeats has not yet been assigned, so the value held in the variable is not known. The later statement `totalHeartbeats = userAgeMinutes * avgBeatsPerMinute;` assigns totalHeartbeats with 378639360.

Feedback?

Limits on int values

In the above example, a userAge value of 57 or greater may yield an incorrect output for totalHeartbeats. The reason is that an int variable can typically only hold values up to about 2 billion; trying to store larger values results in "overflow". Other sections discuss overflow as well as other data types that can hold larger values.