Informatik - AS21

Exercise 4: Loops and Floats

Handout: Oct 11, 2021 6:00 AM

Due: Oct 18, 2021 6:00 PM

Task 1: Loop mix-up: Snippet 1

Open Task

This task is a text based task. You do not need to write any program/C++ file: the answer should be written in main.md (and might include code fragments if questions ask for them).

Task

Considering the snippet

```
unsigned int n;
std::cin >> n;
unsigned int f = 1;
if (n != 0) {
  do {
    f = f * n;
    --n;
  } while (n > 0);
}
std::cout << f << std::endl;
```

1. Describe what it computes.
2. Decide which of the other two kind of loops would fit better than the one it is currently using, and describe why.
3. Rewrite the snippet into the loop you specified in (2). You can use [sandbox task (1)]https://expert.ethz.ch/solve/rRLTDKB5BQybiC83k#AS21-ifme1-Sandbox_Tasks_for_Ex_3-Sandbox_task_(1)) to test that your code behaves correctly. **WARNING:** the sandbox task is only intended for you to test your code. Its content will **not** be graded **nor** award XP, do not forget to copy the rewritten snippet *here* after testing.

Important: The use of `goto` statements is prohibited.

Task 2: Loop mix-up: Snippet 2

Open Task

This task is a text based task. You do not need to write any program/C++ file: the answer should be written in main.md (and might include code fragments if questions ask for them).

Task

Considering the snippet

```
while (true) {
  int i1, i2;
  std::cin >> i1 >> i2;
  std::cout << i1 + i2 << "\n";
  int again;
  std::cout << "Again?(0/1)\n";
  std::cin >> again;
  if (again == 0) break;
}
```

1. Describe what it computes.

2. Decide which of the other two kind of loops would fit better than the one it is currently using, and describe why.

3. Rewrite the snippet into the loop you specified in (2). You can use sandbox task (2) to test that your code behaves correctly.
   WARNING: the sandbox task is only intended for you to test your code. Its content will **not** be graded **nor** award XP, do not forget to copy the rewritten snippet *here* after testing.

Task 3: Loop mix-up: Snippet 3

Open Task

This task is a text based task. You do not need to write any program/C++ file: the answer should be written in main.md (and might include code fragments if questions ask for them).

Task

Considering the snippet

```
unsigned int z;
unsigned int d;

for (std::cin >> z >> d; z >= d; z = z-d);
std::cout << z << std::endl;
```

1. Describe what it computes.
2. Decide which of the other two kind of loops would fit better than the one it is currently using, and describe why.
3. Rewrite the snippet into the loop you specified in (2). You can use sandbox task (3) to test that your code behaves correctly.
   WARNING: the sandbox task is only intended for you to test your code. Its content will **not** be graded **nor** award XP, do not forget to copy the rewritten snippet *here* after testing.

Task 4: Loop Analysis

Open Task

This task is a text based task. You do not need to write any program/C++ file: the answer should be written in main.md (and might include code fragments if questions ask for them).

Task

Consider the following program:

```
unsigned int n;
std::cin >> n;

unsigned int x = 1;
if (n > 0) {
  unsigned int k = 0;
  bool e = true;
  do {
    if (++k == n) {
      e = false;
    }
    x *= 2;
  } while (e);
}
std::cout << x << std::endl;
```

1. Describe the output of the program as a function of its input n.
2. For which values of n do you expect a correct output x? Explain why.
3. Show that this program terminates for all values of n found in (2). Hint: Make an argument based on induction.
4. Provide a more elegant implementation of this function using another type of loop. You can use sandbox task (4) to test that your code behaves correctly.
   WARNING: the sandbox task is only intended for you to test your code. Its content will **not** be graded **nor** award XP, do not forget to copy the rewritten snippet *here* after testing.

Task 5a: Approximation of Pi: Sum 1

Open Task

Task

The number π can be defined through various infinite sums. The accuracy increases with the number of terms. Considering the follwowing sum, that we call sum 1:

$$\frac{\pi}{4} = \sum_{0 \leq n} \frac{(-1)^n}{2n+1} = 1 - \frac{1}{3} + \frac{1}{5} - \frac{1}{7} + \ldots$$

Write a program that computes and outputs an approximation of Pi, based on sum 1. The input for your program is the number of terms m to be included in the calculation to be done. The output is the approximation of π.

Input

A number $m \geq 0$.

Output

The approximation of π given by $4 \sum_{0 \leq n < m} \frac{(-1)^n}{2n+1}$, rounded to 6 significant digits. Note that 6 significant digits is the default precision of C++ for printing floating-point values. Use a variable of type `double` to calculate the sum. Note that that x^0 is 1 (if $x \neq 0$).

Task 5b: Approximation of Pi: Sum 2

Open Task

Task

The number π can be defined through various infinite sums. The accuracy increases with the number of terms. Considering the following sum, which we call sum 2:

$$\begin{aligned} \frac{\pi}{2} &= 1 + \sum_{1 \leq n} \frac{\prod_{0<i\leq n} i}{\prod_{0<i\leq n} (2i+1)} \\ &= 1 + \frac{1}{3} + \frac{1\times2}{3\times5} + \frac{1\times2\times3}{3\times5\times7} + \frac{1\times2\times3\times4}{3\times5\times7\times9} + \ldots \end{aligned}$$

Write a program that computes and outputs an approximation of Pi, based on sum 2. The input for your program is the number of **terms** m to be included in the calculation to be done. The output is the approximation of π.

Optional: After you have solved this and the previous task, think about which formula gives a more precise approximation of $\pi`, sum 1 or sum 2 ? What are the drawbacks? Write your thoughts in a comment below the code.

Input

A natural number m.

Output

The approximation of π given by m terms, rounded to 6 significant digits. Note that 6 significant digits is the default precision of C++ for printing floating-point values. Use a `double` variable to store the sum.