Stepwise improvement sequences with hints

Minor errors by teachers are corrected

Program 1

Example 1 (participant #163020)

	Program state	Hint

	`int sum = 0;`
	`for (int i = 0; i < values.length; i++){`
	`  if (positivesOnly == true) {`
	`  if (values[i] >= 0) {`
	`  sum += values[i];`
	`  }`
	`  } else {`
	`  sum += values[i];`
	`  }`
	`}`
	`return sum;`
		could '== true', at the end of a boolean expression, be removed always?

→	`int sum = 0;`
	`for (int i = 0; i < values.length; i++){`
	`  if (positivesOnly) {`
	`  if (values[i] >= 0) {`
	`  sum += values[i];`
	`  }`
	`  } else {`
	`  sum += values[i];`
	`  }`
	`}`
	`return sum;`
		Would iterating over values be more readable? (f.i. 'for (int i : values) {...')

→	`int sum = 0;`
	`for (int i : values){`
	`  if (positivesOnly) {`
	`  if (i >= 0) {`
	`  sum += i;`
	`  }`
	`  } else {`
	`  sum += i;`
	`  }`
	`}`
	`return sum;`
		Does it make sense to add 0 to a number?

→	`int sum = 0;`
	`for (int i : values) {`
	`  if (positivesOnly) {`
	`  if (i > 0) {`
	`  sum += i;`
	`  }`
	`  } else {`
	`  sum += i;`
	`  }`
	`}`
	`return sum;`
		We have duplication, in 'sum += values[i]'. How could we eliminate it, using a binary operator connecting the 2 boolean expressions? Would this make your code more readable or less? Would this make your code more maintainable or less?

→	`int sum = 0;`
	`for (int i : values){`
	`  if (i > 0 || !positivesOnly) {`
	`  sum += i;`
	`  }`
	`}`
	`return sum;`
		Done

Program 2a

Example 1 (participant #162208)

Program state	Hint

`int total = 0;`
`boolean stop = false;`
 	
`for (int i = 1; i < array.length; i = i + 2) {`
`  if (stop == false) {`
`  if (array[i] >= 0) {`
`  total += array[i];`
`  } else if (array[i] < 0) {`
`  stop = true;`
`  }`
`  }`
`  else {`
`  total = total;`
`  }`
`}`
`return total;`
	Remove lines that does not change the current state (total = total)

`int total = 0;`
`boolean stop = false;`
 	
`for (int i = 1; i < array.length; i = i + 2) {`
`  if (stop == false) {`
`  if (array[i] >= 0) {`
`  total += array[i];`
`  } else if (array[i] < 0) {`
`  stop = true;`
`  }`
`  }`
`}`
`return total;`
	Use break or return instead of the boolean stop

`int total = 0;`
 	
`for (int i = 1; i < array.length; i = i + 2) {`
`  if (array[i] >= 0) {`
`  total += array[i];`
`  } else if (array[i] < 0) {`
`  break;`
`  }`
`}`
`return total;`
	Replace the else if with else, since it always happens

`int total = 0;`
 	
`for (int i = 1; i < array.length; i = i + 2) {`
`  if (array[i] >= 0) {`
`  total += array[i];`
`  } else {`
`  break;`
`  }`
`}`
`return total;`
	Reduce the number of nested statements.

`int total = 0;`
 	
`for (int i = 1; i < array.length && array[i] >= 0; i = i + 2) {`
`  total += array[i];`
`}`
`return total;`
	[introduced a method for isPositiveNumber]

`int total = 0;`
 	
`for (int i = 1; i < array.length && isPositiveNumber(array[i]); i = i + 2) {`
`  total += array[i];`
`}`
`return total;`
	Done

Example 2 (id #105352)

Program state	**Hint**

`int total = 0;`
`boolean stop = false;`
 	
`for (int i = 1; i < array.length; i = i + 2) {`
`  if (stop == false) {`
`  if (array[i] >= 0) {`
`  total += array[i];`
`  } else if (array[i] < 0) {`
`  stop = true;`
`  }`
`  }`
`  else {`
`  total = total;`
`  }`
`}`
`return total;`
	line 13: does nothing

`int total = 0;`
`boolean stop = false;`
 	
`for (int i = 1; i < array.length; i = i + 2) {`
`  if (stop == false) {`
`  if (array[i] >= 0) {`
`  total += array[i];`
`  } else if (array[i] < 0) {`
`  stop = true;`
`  }`
`  }`
`}`
`return total;`
	never use (expression == false), use (!expression) instead

`int total = 0;`
`boolean stop = false;`
 	
`for (int i = 1; i < array.length; i = i + 2) {`
`  if (!stop) {`
`  if (array[i] >= 0) {`
`  total += array[i];`
`  } else if (array[i] < 0) {`
`  stop = true;`
`  }`
`  }`
`}`
`return total;`
	line 8: is this boolean expression useful?

`int total = 0;`
`boolean stop = false;`
 	
`for (int i = 1; i < array.length; i = i + 2) {`
`  if (!stop) {`
`  if (array[i] >= 0) {`
`  total += array[i];`
`  } else  {`
`  stop = true;`
`  }`
`  }`
`}`
`return total;`
	You can add the stop-criterion to the for-statement

`int total = 0;`
`boolean stop = false;`
 	
`for (int i = 1; i < array.length && array[i] >= 0; i = i + 2) {`
`  total += array[i];`
`}`
`return total;`
	Done

Program 2b

Example 1 (participant #111046)

Program state	Hint

`int answer = 0;`
`int index = 0;`
`boolean value = true;`
 	
`for(int number: array) {`
`  if(index % 2 == 0) {`
`  index++;`
`  } else {`
`  if(number == -1) {`
`  value = false;`
`  }`
`  if(value) {`
`  answer = answer + number;`
`  }`
`  index++;`
`  }`
`}`
`return answer;`
	If you are keeping an index for the even/uneven numbers, wouldn't it be easier to use a regular FOR loop instead of a FOREACH?

`int answer = 0;`
`boolean value = true;`
 	
`for(int index = 1; index < array.length;`
`index += 2) {`
`  if(array[index]== -1) {`
`  value = false;`
`  }`
`  if(value) {`
`  answer = answer + array[index];`
`  }`
`}`
`return answer;`
	If we encounter a -1, can't we just return the answer?

`int answer = 0;`
`boolean value = true;`
 	
`for(int index = 1; index < array.length;`
`index += 2) {`
`  if(array[index]== -1) {`
`  return answer;`
`  }`
`  if(value) {`
`  answer = answer + array[index];`
`  }`
`}`
`return answer;`
	If the number is not -1, can't we just assume we can add the number to the sum?

`int answer = 0;`
 	
`for(int index = 1; index < array.length;`
`index += 2) {`
	Can we rewrite x = x + y to something shorter?

`        if(array[index]== -1) {`
`            return answer;`
`        }else {`
`            answer = answer + array[index];`
`        }`
`}`
`return answer;`	

`int answer = 0;`
 	
`for(int index = 1; index < array.length;`
`index += 2) {`
`        if(array[index]== -1) {`
`            return answer;`
`        }else {`
`            answer += array[index];`
`        }`
`}`
`return answer;`
	[Done]

Example 2 (participant # 144022)

Program state	**Hint**

`int answer = 0;`
`int index = 0;`
`boolean value = true;`
 	
`for(int number: array) {`
`    if(index % 2 == 0) {`
`        index++;`
`    } else {`
`        if(number == -1) {`
`            value = false;`
`        }`
`        if(value) {`
`            answer = answer + number;`
`        }`
`        index++;`
`    }`
`}`
`return answer;`
	Factor out index++ and negate condition

`int answer = 0;`
`int index = 0;`
`boolean value = true;`
 	
`for(int number: array) {`
`    if(index % 2 == 1) {`
`        if(number == -1) {`
`            value = false;`
`        }`
`        if(value) {`
`            answer = answer + number;`
`        }`
`    }`
`    index++;`
	Use a "classical" for-loop

} return answer;	

int answer = 0;
boolean value = true;
 	
for(int index = 0; index < array.length;
index++) {
    int number = array[index];
    if(index % 2 == 1) {
        if(number == -1) {
            value = false;
        }
        if(value) {
            answer = answer + number;
        }
    }
}
return answer;
	Redundant variable value (poorly chosen name btw)

int answer = 0;
 	
for(int index = 0; index < array.length;
index++) {
    int number = array[index];
    if(index % 2 == 1) {
        if(number == -1) {
            return answer;
        }
        answer = answer + number;
    }
}
return answer;
	Cosmetics

int answer = 0;
 	
for (int i = 0; i < array.length; i++) {
    int number = array[i];
    if (i % 2 == 1) {
        if (number == -1) {
            break;
        }
        answer += number;
    }
}
return answer;
	[Done]