CSCI 1300 CS1: Starting Computing

Naidu/Correll/Yeh/Hoefer - Fall 2021

Recitation 5 - the week of September 20th, 2021

# Loops

Last week in lecture we learned about using loops: while, do while and for loops. Let’s get some practice with them today.

# Exercises

1. Spot the errors

#include iostream

using namespace std;

int main();

{   
 int x = 10;

int sum = 0;

while(x >= 10)

{

sum += x;

x--;

}

cout << “Sum = “ << sum << endl;

return 0;

}

#include <iostream>

int main()

{   
 int x = 10

int sum = 0

for(int i=0; i=x; i++);

{

sum += x;

}

std::cout << “Sum = “ << sum << std::endl;

return 0;

}

#include <iostream>

using namespace std;

int main();

{   
 int x = 10;

int y = 5;

int sum = 0;

do

{

sum = x + y;

cout << “Sum = “ << sum << end;

x++;

y++;

} while(sum <= 20)

return 0;

}

#include iostream

using namespace std

int main()

{   
 int x = 10;

int y = 5;

int sum = 0;

while(sum <= 20);

{

sum = x + y;

cout << “Sum = “ << sum << endl;

x++;

y++;

}

return 0;

}

#include <iostream>

int main()

{   
 int x = 10;

int sum = 0;

for(i=0; i < x; i++)

{

sum += x

}

cout << “Sum = “ << sum << std::endl;

return 0;

}

#include istream

int main()

{   
 int x = 10;

int sum = -40;

int i;

fore(i=0; i < x; i)

<

sum += x;

>

std::cout << “Sum = “ << sum << std::endl;

return 0;

}

**2.** Convert the following *for* loop into a *while* loop:

#include<iostream>

using namespace std;

int main()

{   
 int x = 10;

int sum = 0;

for(int i=0; i < x; i++)

{

sum += x;

}

cout << “Sum = “ << sum << endl;

return 0;

}

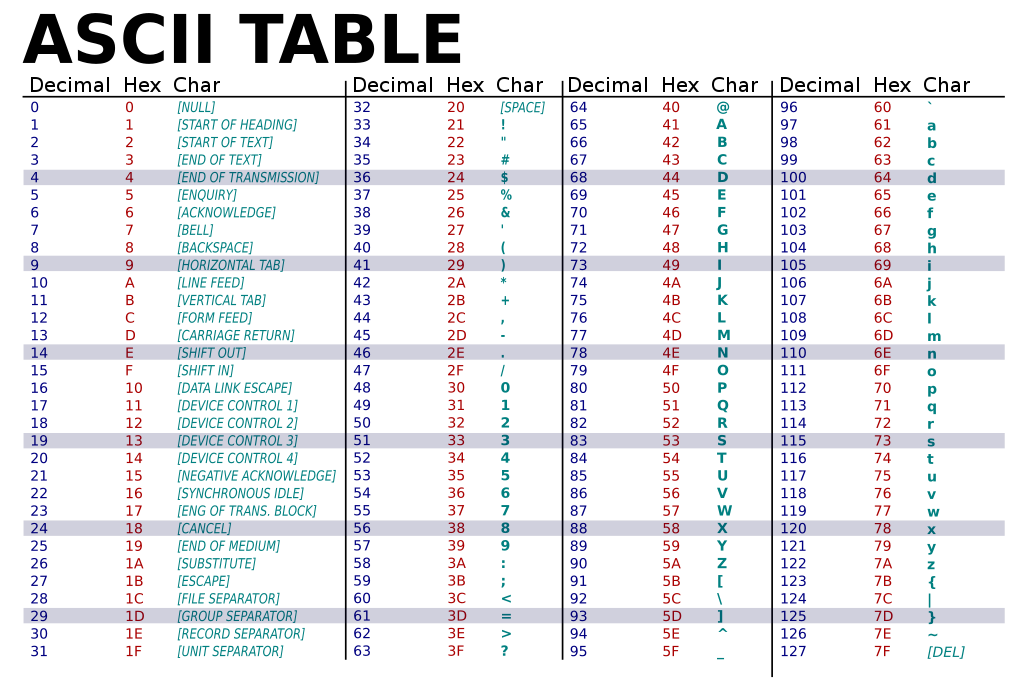
**3. Converting all uppercase letters in a string to lowercase letters.**

The program should ask the user to input a string and convert any uppercase letters in that string to lowercase letters. The program should then print the newly converted string in the console.

Example output (**bold** is user input)

| Please enter a string:  **Welcome to CSCI 1300!**  welcome to csci 1300! |
| --- |

This ASCII table can be used as a reference on the mapping between characters (char) and integers. Test it out yourself: what is the value of int(‘a’)? Note that the difference in value between an uppercase character and its lowercase counterpart is 32.

****

**3a.** Write an algorithm in pseudocode for the program above.

| **Pseudocode:** |
| --- |

**3b.** Let’s draw a flowchart of the solution

| **Flowchart:** |
| --- |

**3c.** Imagine how a sample run of your program would look like. Think about at least two examples

| **Sample Run 1:** |
| --- |

| **Sample Run 2:** |
| --- |

| **Sample Run 3:** |
| --- |

**3d.** Identify the values that you must test for. We call these values “**boundary conditions**”.

| **Answer:** |
| --- |

**3e.** Implement your solution in C++ using VS Code. Revise your solution, save, compile and run it again. Are you getting the expected result and output? Keep revising until you do. Make you sure you test for the values used in your sample runs, and for the boundary conditions.

**4. Removing every other letter from a string**

Let’s create a program that takes a string as user input and prints out the string with every other character removed. The resulting string will keep the first character, omit the second, keep the third, and so on.

Example output (**bold** is user input)

| Please enter a string:  **banana**  bnn |
| --- |

**4a**. Write an algorithm in pseudocode for the above program.

| **Pseudocode:** |
| --- |

**4b.** Draw a flowchart for your solution.

| **Flowchart:** |
| --- |

**4c.** Imagine how a sample run would look like for your program.

| **Sample run 1:** |
| --- |

| **Sample run 2:** |
| --- |

**4d.** Identify the values that you must test for. We call these values “**boundary conditions**”.

| **Answer:** |
| --- |

**4e.** Implement your solution in C++ using VS Code. Revise your solution, save, compile and run it again. Are you getting the expected result and output? Keep revising until you do. Make you sure you test for the values used in your sample runs, and for the boundary conditions.

**5. Printing an equilateral right triangle**

Let’s create a program that takes a side length as user input and prints out a right equilateral triangle of asterisks (“\*”) with that side length. Non-positive side lengths should print “Invalid Input” and return, while a side length of zero should print nothing (i.e. a triangle with side length zero).

Example output (**bold** is user input)

| Enter the triangle side length:  **7**  \*  \*\*  \*\*\*  \*\*\*\*  \*\*\*\*\*  \*\*\*\*\*\*  \*\*\*\*\*\*\* |
| --- |

**5a**. Write an algorithm in pseudocode for the above program.

| **Pseudocode:** |
| --- |

**5b.** Draw a flowchart for your solution.

| **Flowchart:** |
| --- |

**5c.** Imagine how a sample run would look like for your program.

| **Sample run 1:** |
| --- |

| **Sample run 2:** |
| --- |

| **Sample run 3:** |
| --- |

**5d.** Identify the values that you must test for. We call these values “**boundary conditions**”.

| **Answer:** |
| --- |

**5e.** Implement your solution in C++ using VS Code. Revise your solution, save, compile and run it again. Are you getting the expected result and output? Keep revising until you do. Make you sure you test for the values used in your sample runs, and for the boundary conditions.