

## CSE278: Introduction to Systems Programming (Systems I)

### Homework #7

**Due: Wednesday April 22, 2020 before 11:59 PM**

**Email-based help Cutoff: 5:00 PM on Mon, April 20, 2020**

Maximum Points: 50

#### Submission Instructions

This part of the homework assignment must be turned-in electronically via Canvas. Ensure you name this document `HW7_MUID.docx`, where `MUID` is your Miami University unique ID. (Example: `HW7_ahmede.docx`)

Copy pasting from online resources is **Plagiarism**. Instead you should read, understand, and use your own words to respond to questions.

#### Submission Instructions:

Once you have completed answering the questions save this document as a PDF file (**don't just rename the document; that is not the correct way to save as PDF**) and upload it to Canvas.

**General Note:** Upload each file associated with homework (or lab exercises) individually to Canvas. Do not upload archive file formats such as zip/tar/gz/7zip/rar etc.

#### Objective

The objective of this homework is to review basic concepts of:

- Basics of Number Systems
- Basics of computer architecture and organization
- Use CODE Plugin as part of functional testing

Name: **Ben Hilger**

#### Required reading

- Lecture Slides NumberRepresentation
- Lecture Slides ComputerArchitecture

1. Convert hexadecimal 765F to octal. (Hint: First convert 765F to binary, then convert that binary number to octal). *Show all of your mathematical work.*

Hexadecimal: 765F

7: 0111

6: 0110

5: 0101

F (15): 1111

Binary: 0111011001011111

000: 0

111: 7

011: 3

001: 1

011: 3

111: 7

Octal: 73137

2. Convert decimal 299 to binary, to octal and to hexadecimal. *Show all of your mathematical work.*

$$2^8 = 256$$

$$2^9 = 512$$

**\*\* Start with  $2^8$  \*\***

$$299 - 2^8 = 43$$

$$43 - 2^5 = 11$$

$$11 - 2^3 = 3$$

$$3 - 2^1 = 1$$

$$1 - 2^0 = 0$$

Binary Number: 100101011

To Octal:

$$100 = 4$$

$$101 = 5$$

$$011 = 3$$

Octal Number: 453

To Hex:

$$0001: 1$$

$$0010: 2$$

$$1011: B (11)$$

Hexadecimal: 12B

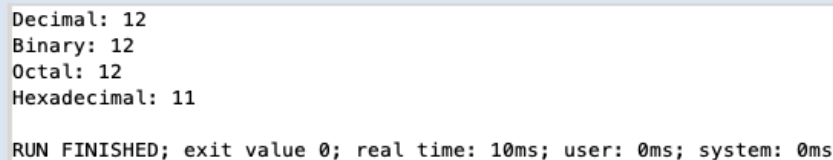
### 3. Literals in C++

C++ supports all 4 number systems to represent numbers: – Normally numbers are treated as decimals – If number starts with **0b** it is considered binary – If number starts with **0** it is considered octal – If number starts with **0x** it is considered hexadecimal. Print the value of the following code snippet:

```
int main() {  
    int i = 12;    // decimal  
    int j = 0b1100; // 12 in binary  
    int k = 014;   // 12 in octal  
    int l = 0xb;   // 11 in hexadecimal  
}
```

Print the values from main suitably calling `cout`

Attach a screen shot of the result



```
Decimal: 12  
Binary: 12  
Octal: 12  
Hexadecimal: 11  
  
RUN FINISHED; exit value 0; real time: 10ms; user: 0ms; system: 0ms
```

### 4. Number I/O in different bases

Run the following code:

```
int main() {  
    int num = 0;  
    std::cout << "Enter hexadecimal number: ";  
    std::cin >> std::hex >> num;  
    // Print number in various bases  
    std::cout << "Decimal: " << std::dec << num  
        << "\nOctal: " << std::oct << num  
        << "\nHexadecimal: " << std::hex << num  
        << std::endl;  
}
```

Attach a screen shot of the result

```
Enter a hexadecimal number: 10A
Decimal: 266
Octal: 412
Hexadecimal: 10a
RUN FINISHED; exit value 0; real time: 5s; user: 0ms; system: 0ms
```

5. Briefly (max 3 sentences each) describe the functionality of following key components of the Central Processing Unit (CPU) [2 points]

- **ALU:** ALU stands for Arithmetic & Logic Unit and performs all of the operations of the CPU. A CPU can have multiple ALU's.
- **Registers:** Registers are the fastest temporary storage locations and are variables designed to use by instructions for the ALU to complete its operations.
- **Caches:** This a special high-speed memory that's usually smaller than RAM and stores a copy of a subset of the data in RAM to ensure fast access.

6. What is the "Von Neumann" or "Stored Program" architecture? [1 point]

The "Von Neumann" architecture is a concept where data and machine language instructions are stored on the same memory. Since they are on the same memory, there is no clear indication of which is which and instead the distinction is made based on how the microprocessor is instructed to interpret the memory.

7. What is a memory address? [1 points]

A memory address is usually a hexadecimal number that is an index in the RAM that points to a byte of data. Consecutive memory address can represent one variable.

8. What is assembly language? State 1 advantage and 1 disadvantage of assembly language [2 points]

Assembly language is a programming language that uses variables to remember a pattern in order to remember values and memory address. This is commonly used for device level programming.

One advantage of assembly language is you can access special features of the processor and it allows you to Ultra-optimize programs for the hardware they are using.

However, one disadvantage that they have is that it's extremely hard to document and maintain and is a lot harder to understand compared to higher-level languages such as Java or C++.

9. Write a complete C++ program which reads one *hexadecimal* number from user and print its equivalent *decimal* number. You should write a function **hex2dec** which can take one argument **command line parameter argv[1]** as string that can represent the hex number. The function should define *int* variable and return the value of the variable as the decimal number. As for your hints, I am attaching the code **bin2Dec()** that we covered earlier in the Lab0:

```
int bin2dec (int binaryNum) {  
    int decimalNum = 0;  
    int remainder;  
    int counter = 0;  
  
    while (binaryNum > 0){  
        remainder = binaryNum % 2;  
        decimalNum += remainder * pow(2, counter);  
        counter++;  
    }
```

```
        binaryNum /= 10;
    }
    return decimalNum;
}
```

The process of computing the decimal number from hexadecimal is similar, here the power to be used is 16. You need to know the length of the string that will be used as the highest power, then inside a *for/while* loop, accumulate the equivalent multiplied by the respective power of 16.

Functional Requirements of the Code are as follows:

- main() function should be able to handle *one* command line string parameter
- Declare a function prototype: `int hex2Dec(string input)`
- Implement the function (use `stoi()` and other string processing API)
- Call the implemented function from `main()`
- Print the result using `std::cout`

### SAMPLE I/O

```
./MyHex2Dec 123
291
```

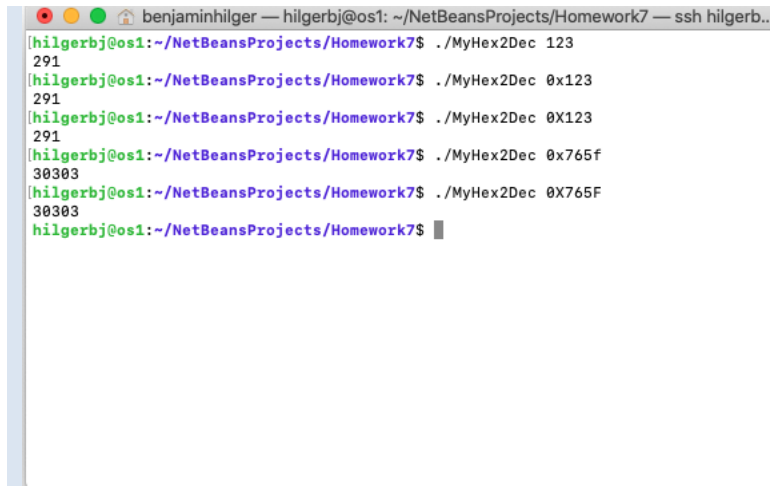
```
./MyHex2Dec 0x123
291
```

```
./MyHex2Dec 0X123
291
```

```
./MyHex2Dec 0x765f
30303
```

```
./MyHex2Dec 0X765F
30303
```

Attach a screen shot of sample run



```
benjaminhilger — hilgerbj@os1: ~/NetBeansProjects/Homework7 — ssh hilgerb...
[hilgerbj@os1:~/NetBeansProjects/Homework7$ ./MyHex2Dec 123
291
[hilgerbj@os1:~/NetBeansProjects/Homework7$ ./MyHex2Dec 0x123
291
[hilgerbj@os1:~/NetBeansProjects/Homework7$ ./MyHex2Dec 0X123
291
[hilgerbj@os1:~/NetBeansProjects/Homework7$ ./MyHex2Dec 0x765f
30303
[hilgerbj@os1:~/NetBeansProjects/Homework7$ ./MyHex2Dec 0X765F
30303
[hilgerbj@os1:~/NetBeansProjects/Homework7$ ]
```

## Submission

- No late assignments will be accepted!
- This work is to be done individually
- The submission file will be saved with the name ***HW7\_yourMUID.pdf***
- Assignment is due before Midnight Wednesday April 22, 2020.
- On or before the due time, drop the *electronic copy* of your work in the *canvas*
- **Don't forget to Turn in the files! HW7\_yourMUID.pdf & HW7\_yourMUID\*.cpp**
- **Also please submit the complete code solution for Q9 via CODE plugin**