Problem Set 1 - Data Structures & Algorithms

22/02/2024

Aditya Narayan Rai - 235843

1. Convert the following decimal integers into 8-bit Two’s Complement signed integers. If that isn’t possible, explain why it isn’t.
   A. 24
   B. 156
   C. -37
   D. 16

Solutions

A. 24 - Since 24 is a positive integer and within the range of -128 to 127, it's binary representation remains the same when converted to the 8-bit Two's Complement signed integers: 00011000

B. 156 - 156 cannot be converted into 8-bit Two's Complement signed integers because an 8-bit Two’s Complement signed integer can represent values from -128 to 127.

C. -37 - Since -37 is a negative integer and within the range of -128 to 127, therefore to convert it to an 8-bit Two’s Complement signed integer, we need to do these things: find the binary representation of 37, flip the binary representation, and finally add 1 to the resulting binary representation. The binary representation of 37 is 00100101. On flipping this we get 11011010. And, finally after adding 1 we get the 8-bit Two’s Complement signed integer which is 11011011.

D. 16 - Since 16 is a positive integer and within the range of -128 to 127, it's binary representation remains the same when converted to the 8-bit Two's Complement signed integers: 00010000

2. Now do the following addition / subtraction problems, highlighting where overflow / underflowerrors have occured. The integers are provided in Two's Complement notation.
   A. 00010010 + 10011100
   B. 00011000 + 00001000
   C. 01100010 + 00101100
   D. 11001111 + 10101001

Solutions

A. 00010010 + 10011100 - On adding these two binary integers, we get **10101110**. There is no overflow/underflow. In decimal notation, we are adding 18 and -100 and therefore the resulting decimal, -82 is within the range of -128 to 127.

B. 00011000 + 00001000 - On adding these two binary integers, we get **00100000**. There is no overflow/underflow. In decimal notation, we are adding 24 and 8 and therefore the resulting decimal, 32 is within the range of -128 to 127.

C. 01100010 + 00101100 - On adding these two binary integers, we get **010001110**. There is overflow in this case since the resulting binary is above the 8bit limit. In decimal notation, we are adding 98 and 44 and therefore the resulting decimal, 142 is outside the range of -128 to 127.

D. 11001111 + 10101001 - On adding these two binary integers, we get **101111000**. There is underflow in this case since the computation produces a value that falls outside the range of values that can be represented. In decimal notation, the values -49 and -87 are being added together.

3. Convert the following fractions into 8-bit floting point (1 sign bit, 3 exponent bits and 4 mantissa bits). If that isn't possible exactly, explain why, and show how inaccurate the encoding would be:
   A. $3\frac{1}{4}$
   B. $-6\frac{1}{2}$
   C. $4\frac{1}{2}$
   D. $\frac{9}{256}$

Solutions

A. $3\frac{1}{4}$**:** Here, 3 is represented by 11 and $\frac{1}{4}$ is represented by 01. So, the mantissa becomes 11.01. Based on the excess notation we have 110 as our exponent and since it's positive our sign bit is 0. Combining all these will give us our binary representation for the fraction: **01101101**

B. $-6\frac{1}{2}$**:** Here our sign bit is 1 and 6 is represented by 110 and $\frac{1}{2}$ is represented by 1. Therefore our mantissa is 110.1. Now based on the excess notation we have 111 as our exponent. Combining all these will give us our binary representation for the fraction: **11110111**

C. $4\frac{1}{2}$**:** Here 4 is represented by 100 and $\frac{1}{4}$ is represented by 1. So, the mantissa becomes 100.1. Based on the excess notation, we have 111 as our exponent and since it's positive sign bit is 0. Combining all these will give us our binary representation for the fraction: **01111001**

D. $\frac{9}{256}$**:** We can't convert this into 8-bit floating point becuase the exponent bit is too big and we can't represent it by the 3-bit excess notation.

4. Convert the following 8-bit floating numbers into fractions. If that isn't possible, explain why:
    A. 01101011
    B. 11011001
    C. 11001011
    D. 01111111

Solutions

A. 01101011: Following the sign, exponent and mantissa rule, we can convert this in the following way: we have the 0 in the beginning so it's positive, our exponent is 110 which in excess notation is 2, and our mantissa is 1011 which as per the exponent will 10.11. Now on converting this we get: ***$2+\frac{1}{2}+\frac{1}{4} = 3\frac{3}{4}$***

B. 11011001: Following the sign, exponent and mantissa rule, we can convert this in the following way: we have the 1 in the beginning so it's negative, our exponent is 101 which in excess notation is 1, and our mantissa is 1001 which as per the exponent will be 1.001. Now on converting this we get: ***$-(1+\frac{1}{8}) = -1\frac{1}{8}$***

C. 11001011: Following the sign, exponent and mantissa rule, we can convert this in the following way: we have 1 in the beginning so it's negative, our exponent is 100 which in excess notation is 0, and our mantissa is 1011 which as per the exponent will be .1011. Now on converting this we get: ***$-(\frac{1}{2}+\frac{1}{8}+\frac{1}{16}) = -\frac{11}{16}$***

D. 01111111: Following the sign, exponent and mantissa rule, we can convert this in the following way: we have 0 in the beginning so it's positive, our exponent is 111 which in excess notation is 3, and our mantissa is 1111 which as per the exponent will be 111.1. Now on converting this we get: ***$4+2+1+\frac{1}{2} = 7\frac{1}{2}$***

5. Write out exactly what will be done by the computer initialized to have nothing in its registers, A0 in the program counter, and the following data in its main memory: Address Contents A0 14 A1 AA A2 23 A3 AA A4 A3 A5 03 A6 82 A7 34 A8 C0 A9 00 AA B49 00 AA B4

Solutions

1. Program counter: A0

- Fetch: 14AA
- Decode: Now, LOAD the bit pattern B4 into register R4
- Execute: Put 10110100 in R4.

2. Program counter: A2

- Fetch: 23AA
- Decode: Now, LOAD the bit pattern AA into register R3
- Execute: Put 10101010 in R3

3. Program counter: A4

- Fetch: A303
- Decode: Rotate the pattern in register R3 to the right 3 times
- Execute: After three rotations, it becomes 01010101

4. Program counter: A6

- Fetch: 8234
- Deocode: AND the bit patterns in register 3 and 4 and put the result in register 2
- Execute: R2 contains 10110100 AND 01010101. After performing AND, R2 becomes 00010100

5. Program counter: A8

- Fetch: C000
- Decode: HALT
- Execute: HALT

Therefore, register 2 contains 14 - 00010100

6. Write a Python function that checks if an input number is a palindrome number (a number that reads the same forward and reverse, e.g. 151) and returns True or False; this function should raise an error if the input is not an integer.

Solution - Let's write the function below:

```
In [2]: def is_palindrome(x):
            # This checks if the input is an integer
            if not isinstance(x, int):
                raise ValueError("Please enter a whole number")

            # This converts the number to a string
            num_str = str(x)

            # And finally this checks if the string reads the same forward and backward
            return num_str == num_str[::-1]
```

```
In [3]: is_palindrome(221122)
```

```
Out[3]: True
```

```
In [4]: is_palindrome(133)
```

```
Out[4]:  False
```

```
In [5]:  is_palindrome('yes')
```

```
---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
Cell In[5], line 1
----> 1 is_palindrome('yes')

Cell In[2], line 4, in is_palindrome(x)
      1 def is_palindrome(x):
      2     # This checks if the input is an integer
      3     if not isinstance(x, int):
----> 4         raise ValueError("Please enter a whole number")
      6     # This converts the number to a string
      7     num_str = str(x)

ValueError: Please enter a whole number
```

7. Using the flask project that you created in the lab, create an About page with the content listed below. You should show in your code that you know how to pass a variable to the render_template function. You need to submit the code of your about.html template and the code of your flaskapp.py file, plus a screenshot of the about page in your running app. Include the relevant code in a script listing in a markdown cell of the Jupyter notebook by creating a markdown cell and surrounding it with three backticks (```: on a US keyboard, this is above the Tab key, on a German keyboard, this is next to the Backspace key). The screenshot can be inserted using the notebook's UI (in my Jupyter, an image can be embedded using Edit - Insert Image in a Markdown cell).
   - a title, e.g. your name
   - a subtitle, e.g. a description of you
   - a photo of you
   - a paragraph about you
   - links to at least one social media account

Solution: Following is the flaskapp.py code chunk:

```
from flask import Flask, render_template

app = Flask(__name__)


@app.route("/")
@app.route("/home")
def home():
    return render_template('home.html', heading='My personal home
page')
```

```
@app.route("/hello")
@app.route("/hello/<city>")
def hello(city=None):
    return render_template('hello.html', city=city)


@app.route("/about")
def about():
    about_content = dict(title='Aditya Narayan Rai',
                         subtitle='Graduate Student - Data Science
for Public Policy, Hertie School, Berlin',
                         photo_url='https://shorturl.at/doCYZ',
                         paragraph=("Hey there,\n"
                                    "Welcome to my page!\n"
                                    "I am Aditya Narayan Rai and I
am a student of Data Science for Public Policy at \n"
                                    "Hertie School, Berlin. Before
starting my studies at Hertie, I was working as a \n"
                                    "Policy Research Consultant in
India. \n"
                                    "I started working right after
my Master's in Public Policy degree.\n"
                                    "During my work as a Policy
Research Consultant, I worked with four different \n"
                                    "Non-Government Orgnizations in
the field of Education, Livelihoods and Governance.\n"
                                    "Now a days, I am busy learning
the intricacies of the policy research and how data \n"
                                    "can be used for common good
purposes. If you are in and around, you can always find in \n"
                                    "some corner of the cafeteria
and/or the library. Oh yes, I am also reading about \n"
                                    "politics and elections and
understanding how core concepts of data science can be applied\n"
                                    "to the field therefore I am
always up for talking and discussion on the topic."),

twitter_link='https://twitter.com/adi1709_aditya',

linkedIn_link='https://www.linkedin.com/in/aditya-narayan-rai/',

github_link='https://github.com/adityanarayan-rai')
    return render_template('about.html',
about_content=about_content)


if __name__ == "__main__":
    app.run(debug=True)e)
```

Note to the Professor: For understanding and improving the placing of the contents, I used Google search (stackoverflow) and ChatGPT as reference.

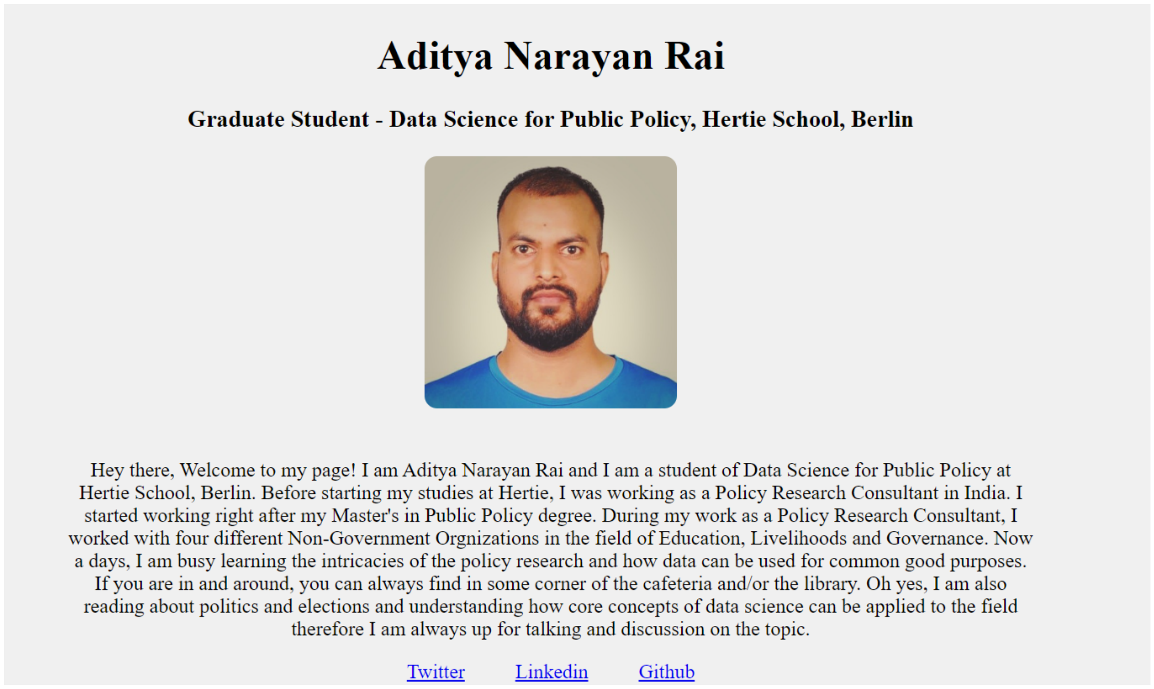
Following is the about.html code chunk:

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-
scale=1.0">
    <title>{{ about_content['title'] }} - About</title>
    <style>
        body {
            display: flex;
            align-items: center;
            justify-content: space-evenly;
            height: 100vh;
            margin: 0;
            background-color: #f0f0f0;
        }
        .content-container {
            text-align: center;
            max-width: 60%;
        img {
            width: 200px;
            border-radius: 10px;
            margin-bottom: 20px;
        }
        ul {
            list-style: none;
            padding: 0;
            margin: 0;
            display: flex;
            justify-content: center;
        }
        li {
            margin-inline: 20px;
        }
    </style>
</head>
<body>
    <div class="content-container">
    <h1>{{ about_content['title'] }}</h1>
    <h3>{{ about_content['subtitle'] }}</h3>
    <img alt="Your Photo" src="{{ about_content['photo_url'] }}"
width="200">
    <p>{{ about_content['paragraph'] }}</p>
    <ul>
            {% for platform, link in about_content.items() if
platform.endswith('_link') %}
                <li><a href="{{ link }}" target="_blank">{{
```

```
platform.split('_')[0].capitalize() }}</a></li>
            {% endfor %}
        </ul>
</body>
</html>l>
```

Note to the Professor: For understanding and improving the placing of the contents, I used Google search (stackoverflow) and ChatGPT as reference.

Following is the screenshot:



Aditya Narayan Rai

Graduate Student - Data Science for Public Policy, Hertie School, Berlin

Hey there, Welcome to my page! I am Aditya Narayan Rai and I am a student of Data Science for Public Policy at Hertie School, Berlin. Before starting my studies at Hertie, I was working as a Policy Research Consultant in India. I started working right after my Master's in Public Policy degree. During my work as a Policy Research Consultant, I worked with four different Non-Government Orgnizations in the field of Education, Livelihoods and Governance. Now a days, I am busy learning the intricacies of the policy research and how data can be used for common good purposes. If you are in and around, you can always find in some corner of the cafeteria and/or the library. Oh yes, I am also reading about politics and elections and understanding how core concepts of data science can be applied to the field therefore I am always up for talking and discussion on the topic.

Twitter Linkedin Github