ч



МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ

“КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ

імені ІГОРЯ СІКОРСЬКОГО”

Факультет прикладної математики

Кафедра програмного забезпечення комп’ютерних систем

**Звіт з лабораторних робіт**

з дисципліни “Програмне забезпечення комп'ютерних систем автоматичної ідентифікації”

|  |  |  |  |
| --- | --- | --- | --- |
| Виконав  студент групи КП-11мп  Перегуда Ярослав Іванович  (*прізвище, ім’я, по батькові*) |  |  | Зарахована  “\_\_\_\_” “\_\_\_\_\_\_\_\_\_\_\_\_” 2021 р.  викладачем  Боярінова Юлія Євгенівна  (*прізвище, ім’я, по батькові*) |

Київ 2021

**Мета роботи**

В результаті циклу лабораторних робіт буде створено програму, що включає в собі метод кодування вхідної інформації, метод друкування (зберігання у виді файлу) штрихової позначки, програму декодування файлу штрихової позначки та вивід результати в інтерфейсі користувача.

**Завдання**

**Варіант 7: UPC (була обрана версія UPC-A)**

**Лабораторна робота №1**. Розробка архітектури бази даних КСАІ.

*Завдання*: розробити архітектуру бази даних КСАІ для збереження інформації про одиниці обліку, розробити СУБД.

**Лабораторна робота №2**. Розробка інтерфейсу користувача КСАІ.

*Завдання*: розробити дизайн та інтерфейс користувача КСАІ.

**Лабораторна робота №3**. Розробка програми кодування вихідної інормації

КСАІ – 6 акад.год.

*Завдання*: розробити алгоритм та програму штрихового кодування інформації

про одиниці обліку (використовуючи результат виконання розрахункової роботи).

**Лабораторна робота №4.** Розробка програми “друкування”(зберігання у вигляді файлу) штрих кодової позначки .

*Завдання*: розробити алгоритм та програму «друкування» штрих кодової

позначки (з урахуванням особливостей одиниці обліку).

**Лабораторна робота №5.** Розробка програми декодування штрих кодової

позначки .

*Завдання*: розробити алгоритм та програму декодування штрих кодової

позначки.

**Лабораторна робота №6.** Розробка та налагодження пакету програм КСАІ .

*Завдання*: розробити та налагодити пакет програм КСАІ.

**Код програмного забезпечення**

1. ***Лістинг коду, який кодує вхідну інформацію у штрих кодову позначку та надає можливість зберегти результуючу картинку у вигляді файлу***

|  |
| --- |
| const encodeCanvas = document.getElementById('encode-barcode');  const encodeCtx = encodeCanvas.getContext('2d');  encodeCtx.imageSmoothingEnabled = false;  const encodeBinary = document.getElementById('encode-binary');  const encodeError = document.getElementById('encode-error');  const encodeResult = document.getElementById('encode-result');  const encodeFromString = () => {  const barcodeStr = document.getElementById('encode-str').value;  encodeUPCBarcode(barcodeStr);  };  const encodeFromInputs = () => {  const price = document.getElementById('price').value;  if (price === '' || +price < 1 || +price > 100) {  encodeError.innerText = 'Invalid price';  return;  }  const rgbArr = hexToRgb(document.getElementById('color').value);  const colorInSixNumbers = rgbArr  .map((number) => {  let dividedNumber = Math.round(+number / 2.55);  if (dividedNumber === 100) {  dividedNumber--;  }  return ('' + dividedNumber).padStart(2, '0');  })  .join('');  const barcodeStr =  document.getElementById('type').value +  document.getElementById('country').value.padStart(2, '0') +  colorInSixNumbers +  price.padStart(2, '0');  encodeUPCBarcode(barcodeStr);  };  const encodeUPCBarcode = (barcodeStr) => {  clearOutput(encodeResult, encodeBinary, encodeCanvas, encodeCtx, encodeError);  if (barcodeStr.length > 12 || barcodeStr.length < 11) {  encodeError.innerText = 'Length must be 11 or 12';  return;  }  const checkSum = getCheckSum(barcodeStr);  if (barcodeStr.length === 11) {  barcodeStr += checkSum;  } else if (+barcodeStr[11] !== checkSum) {  encodeError.innerText = 'CheckSum value (last number) is not valid';  return;  }  const barcodeStrokesArr = getBarcodeStrokesArr(barcodeStr);  drawBarcode(barcodeStrokesArr, barcodeStr, encodeCtx);  manageEncodeOutput(barcodeStrokesArr);  };  const manageEncodeOutput = (barcodeStrokesArr) => {  encodeBinary.value = barcodeStrokesArr.map((item) => item.binary).join(' ');  encodeResult.style.display = 'block';  };  const downloadBarcode = () => {  const link = document.createElement('a');  link.download = 'upc.png';  link.href = encodeCanvas.toDataURL();  link.click();  link.delete;  }; |

1. ***Лістинг коду, який декодує штрих кодову позначку отриману у вигляді строки або файлу***

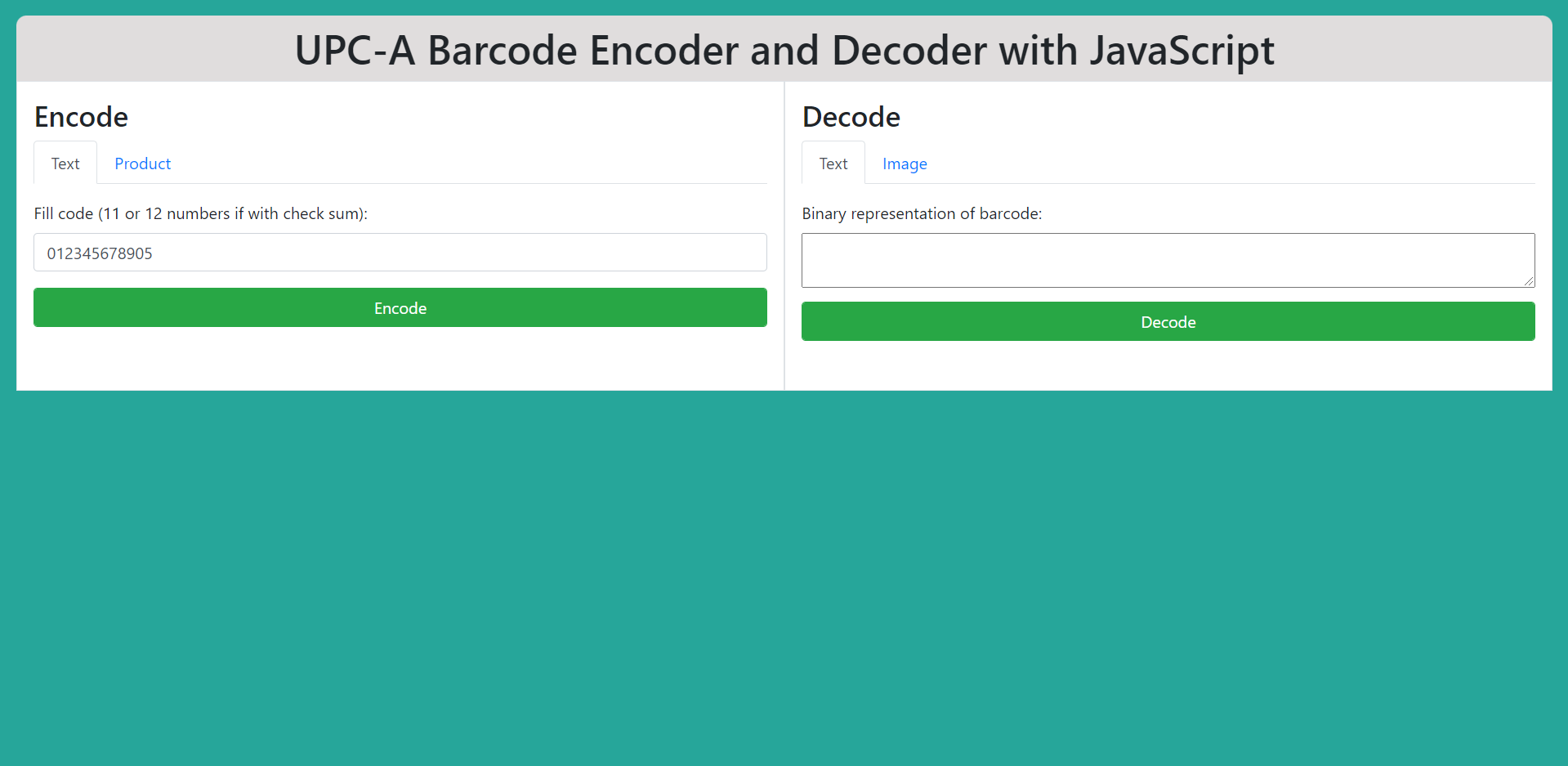
|  |
| --- |
| const decodeCanvas = document.getElementById('decode-barcode');  const decodeCtx = decodeCanvas.getContext('2d');  const decodeStr = document.getElementById('decode-str');  const decodeError = document.getElementById('decode-error');  const decodeResult = document.getElementById('decode-result');  const getKeyByValue = (object, value) => {  return Object.keys(object).find((key) => object[key] === value);  };  const decodeFromString = () => {  const barcodeBinary = document.getElementById('decode-binary').value;  const binaryArr = barcodeBinary.split(' ');  decodeUPCBarcode(binaryArr);  };  const decodeFromImage = async () => {  const fileinput = document.getElementById('decode-file'); // input file  const file = fileinput.files[0];  const img = new Image();  const reader = new FileReader();  // Read in the image file as a data URL.  reader.readAsDataURL(file);  reader.onload = function (evt) {  if (evt.target.readyState === FileReader.DONE) {  img.src = evt.target.result;  img.onload = () => {  decodeCtx.clearRect(0, 0, decodeCanvas.width, decodeCanvas.height);  decodeCtx.drawImage(img, 0, 0, decodeCanvas.width, decodeCanvas.height);  let binaryStr = '';  for (let i = 0; i < 110; i++) {  const pixel = decodeCtx.getImageData(i \* 2, 0, 1, 1);  const data = pixel.data;  binaryStr += data[3] / 255 !== 0 ? 1 : 0;  }  let formatedBinaryStr = '';  let sliceStart = 0;  for (let i = 0; i < 17; i++) {  if (i === 1 || i === 15) {  formatedBinaryStr +=  binaryStr.slice(sliceStart, sliceStart + 3) + ' ';  sliceStart += 3;  continue;  }  if (i === 8) {  formatedBinaryStr +=  binaryStr.slice(sliceStart, sliceStart + 5) + ' ';  sliceStart += 5;  continue;  }  if (i === 16) {  formatedBinaryStr += binaryStr.slice(sliceStart, sliceStart + 7);  sliceStart += 7;  continue;  }  formatedBinaryStr +=  binaryStr.slice(sliceStart, sliceStart + 7) + ' ';  sliceStart += 7;  }  const binaryArr = formatedBinaryStr.split(' ');  decodeUPCBarcode(binaryArr);  };  }  };  };  const getStrFromBinaryArr = (binaryArr) => {  let firstGroup = true;  const numberArr = binaryArr.map((elem) => {  if (elem === silence || elem === start\_stop) {  return '';  }  if (elem === middle) {  firstGroup = false;  return '';  }  if (firstGroup) {  return getKeyByValue(groupA, elem);  } else {  return getKeyByValue(groupC, elem);  }  });  return numberArr.join('');  };  const decodeUPCBarcode = async (binaryArr) => {  clearOutput(decodeResult, decodeStr, decodeCanvas, decodeCtx, decodeError);  const barcodeStr = getStrFromBinaryArr(binaryArr);  const productData = await getDataFromStr(barcodeStr);  if (barcodeStr.length !== 12) {  decodeError.innerText =  'Must be encoded 12 numbers. Currently encoded ' + barcodeStr.length;  return;  }  const barcodeStrokesArr = getBarcodeStrokesArr(barcodeStr);  drawBarcode(barcodeStrokesArr, barcodeStr, decodeCtx);  manageDecodeOutput(barcodeStr, productData);  };  const manageDecodeOutput = (barcodeStr, productData) => {  decodeStr.value = barcodeStr;  decodeStr.style.display = 'block';  document.getElementById('decoded-type').value = productData.type;  document.getElementById('decoded-country').value = productData.country;  document.getElementById('decoded-price').value = productData.price;  document.getElementById('decoded-color').value = rgbToHex.apply(  null,  productData.color  );  decodeResult.style.display = 'block';  }; |

1. ***Лістинг допоміжного коду, який відповідає за шифрування/дешифрування інформації, отримання інформації з бази даних, вимальовування штрих кодової позначки у вигляді картинки, тощо.***

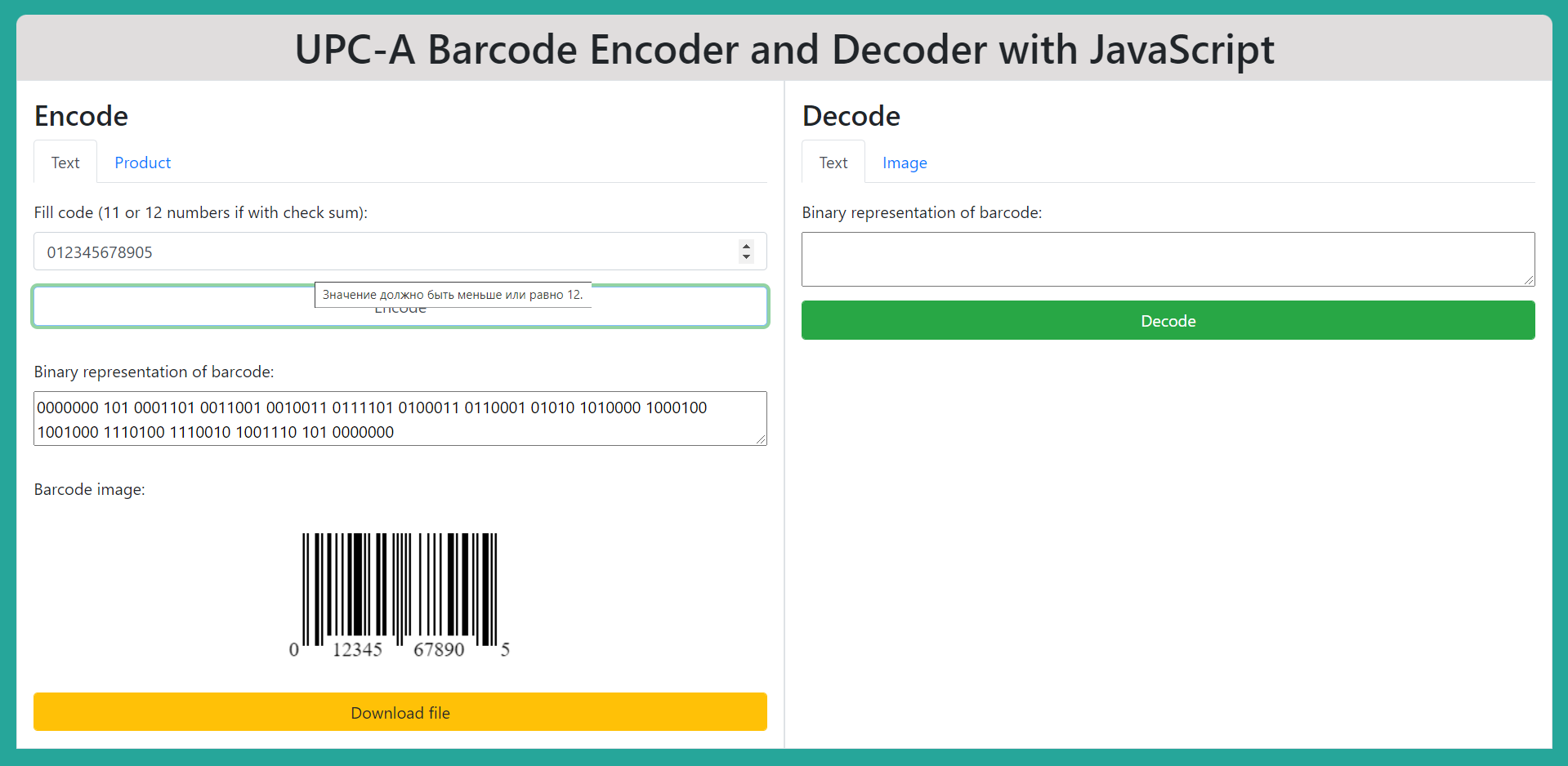
|  |
| --- |
| const groupA = {  0: '0001101',  1: '0011001',  2: '0010011',  3: '0111101',  4: '0100011',  5: '0110001',  6: '0101111',  7: '0111011',  8: '0110111',  9: '0001011'  };  const groupC = {  0: '1110010',  1: '1100110',  2: '1101100',  3: '1000010',  4: '1011100',  5: '1001110',  6: '1010000',  7: '1000100',  8: '1001000',  9: '1110100'  };  const silence = '0000000';  const start\_stop = '101';  const middle = '01010';  const hexToRgb = (hex) =>  hex  .replace(  /^#?([a-f\d])([a-f\d])([a-f\d])$/i,  (m, r, g, b) => '#' + r + r + g + g + b + b  )  .substring(1)  .match(/.{2}/g)  .map((x) => parseInt(x, 16));  const rgbToHex = (r, g, b) =>  '#' +  [r, g, b]  .map((x) => {  const hex = x.toString(16);  return hex.length === 1 ? '0' + hex : hex;  })  .join('');  const getCheckSum = (barcodeStr) => {  let oddSum = 0;  let evenSum = 0;  [...barcodeStr].forEach((char, i) => {  if (i === 11) {  return;  }  i % 2 === 0 ? (evenSum += parseInt(char)) : (oddSum += parseInt(char));  });  return (10 - ((oddSum \* 3 + evenSum) % 10)) % 10;  };  const getBarcodeStrokesArr = (barcodeStr) => {  let barcodeBinary = [  { tall: true, binary: silence },  { tall: true, binary: start\_stop }  ];  [...barcodeStr].forEach((char, i) => {  if (i === 11) {  barcodeBinary.push({ tall: true, binary: groupC[char] });  return;  }  if (i === 0) {  barcodeBinary.push({ tall: true, binary: groupA[char] });  return;  }  if (i === 6) {  barcodeBinary.push({ tall: true, binary: middle });  }  i < 6  ? barcodeBinary.push({ tall: false, binary: groupA[char] })  : barcodeBinary.push({ tall: false, binary: groupC[char] });  });  barcodeBinary.push(  { tall: true, binary: start\_stop },  { tall: true, binary: silence }  );  return barcodeBinary;  };  const drawBarcode = (barcodeBinary, barcodeStr, ctx) => {  let counter = 0;  ctx.translate(0.5, 0);  barcodeBinary.forEach((number) => {  const binary = number.binary;  const additionalHeight = number.tall ? 10 : 0;  [...binary].forEach((char) => {  if (char === '1') {  for (let i = 0; i < 2; i++) {  ctx.beginPath();  ctx.moveTo(counter \* 2 + i, 0);  ctx.lineTo(counter \* 2 + i, 100 + additionalHeight);  ctx.stroke();  }  }  counter++;  });  });  ctx.font = '20px serif';  ctx.fillText(barcodeStr.slice(0, 1), 0, 120);  ctx.fillText(barcodeStr.slice(1, 6), 42, 120);  ctx.fillText(barcodeStr.slice(6, 11), 122, 120);  ctx.fillText(barcodeStr.slice(11, 12), 207, 120);  ctx.translate(-0.5, 0);  };  const clearOutput = (div, resStr, canvas, ctx, error) => {  resStr.value = '';  error.innerText = '';  ctx.clearRect(0, 0, canvas.width, canvas.height);  div.style.display = 'none';  };  const getDataFromStr = async (barcodeStr) => {  return fetch(`/api/barcode-data`)  .then((json) => json.text())  .then((json) => {  const barcodeData = JSON.parse(json).barcodeData;  const colorStr = barcodeStr.slice(3, 9);  return {  type: barcodeData.types[+barcodeStr.slice(0, 1)],  country: barcodeData.countries[+barcodeStr.slice(1, 3)],  color: [  Math.round(+colorStr.slice(0, 2) \* 2.55),  Math.round(+colorStr.slice(2, 4) \* 2.55),  Math.round(+colorStr.slice(4, 6) \* 2.55)  ],  price: barcodeStr.slice(9, 11)  };  });  }; |

**Демонстрація роботи розробленого програмного забезпечення**

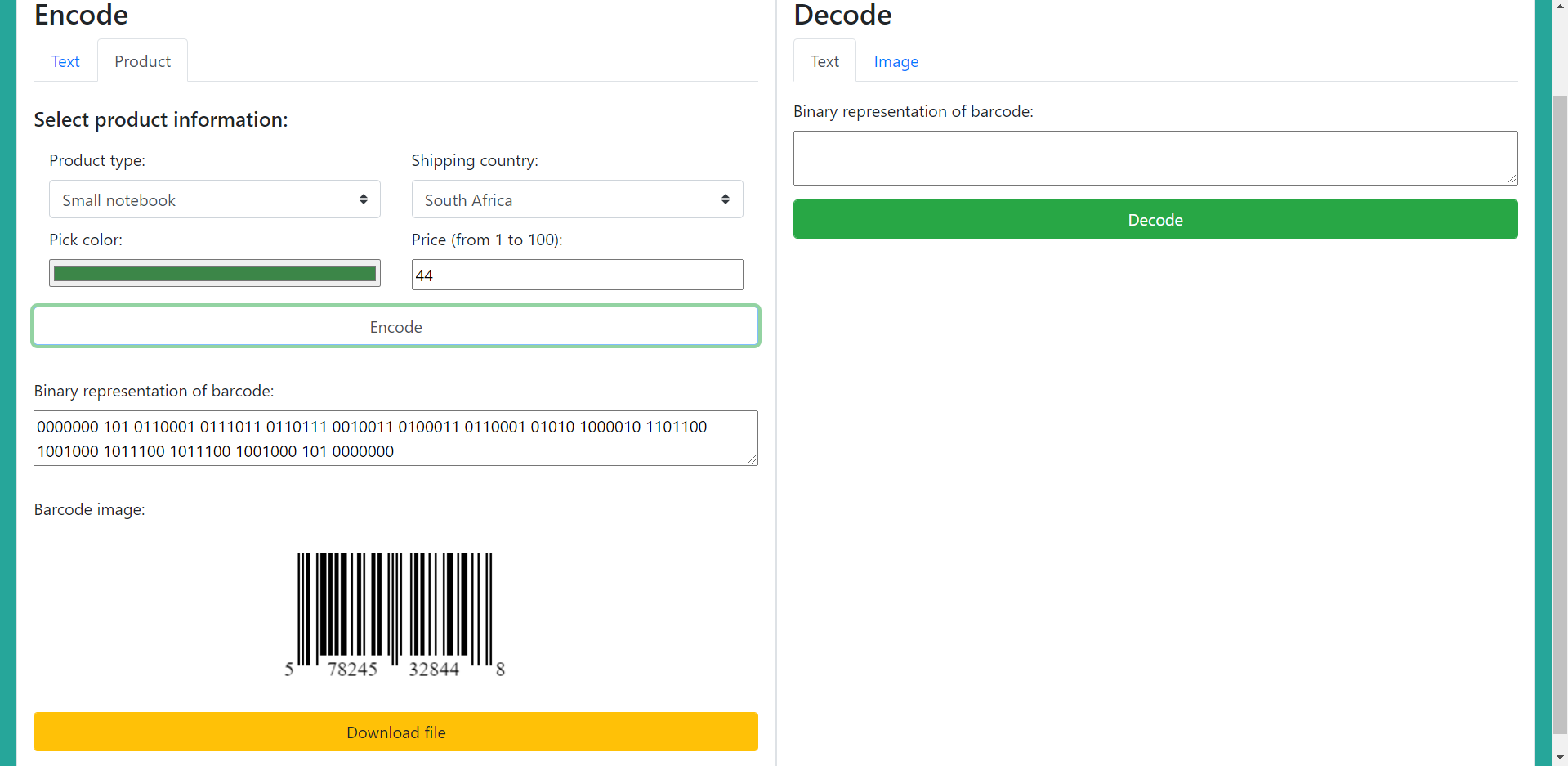
* Графічний інтерфейс застосунку



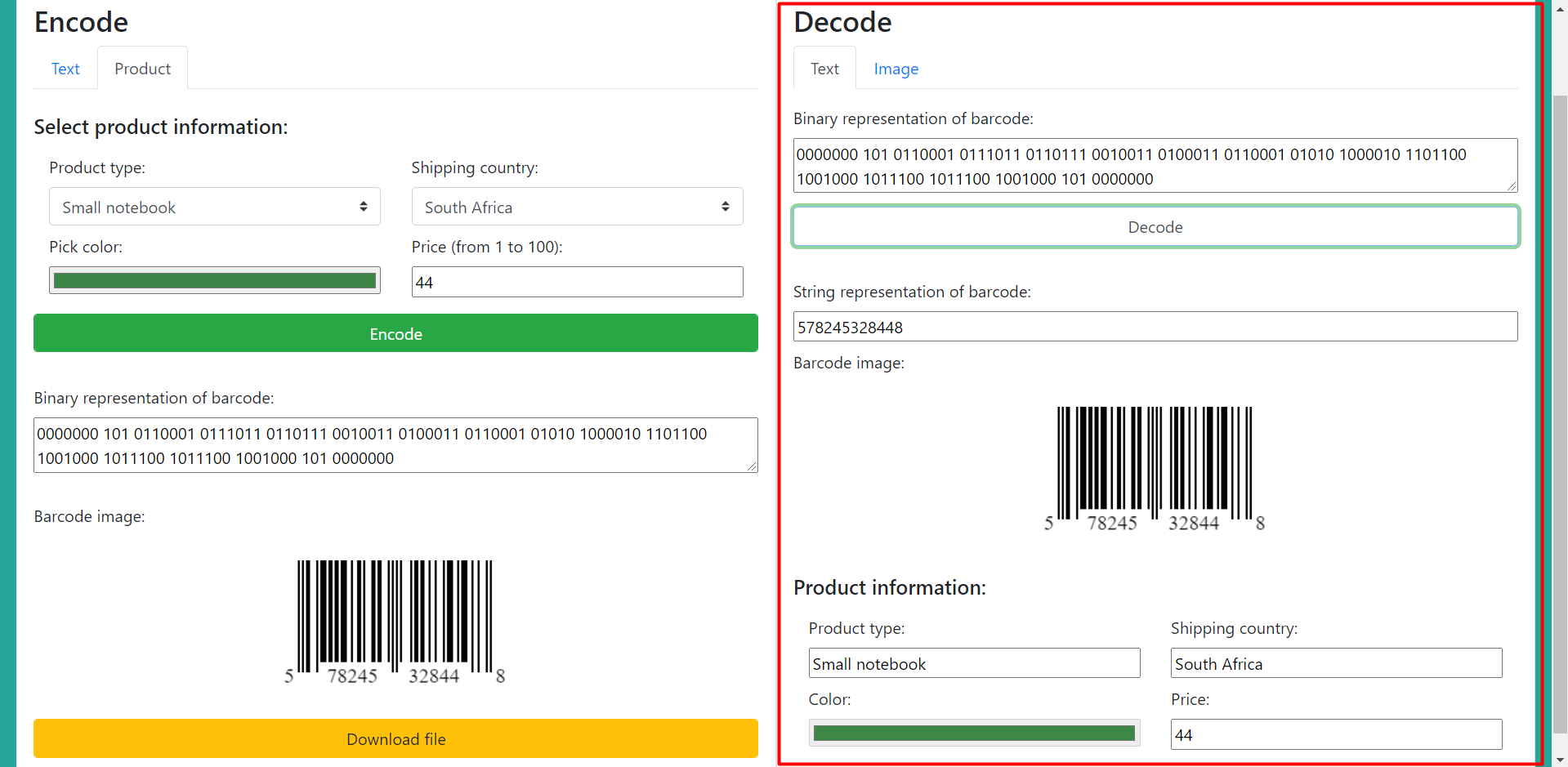
* Кодування вхідної інформації отриманої у вигляді строки



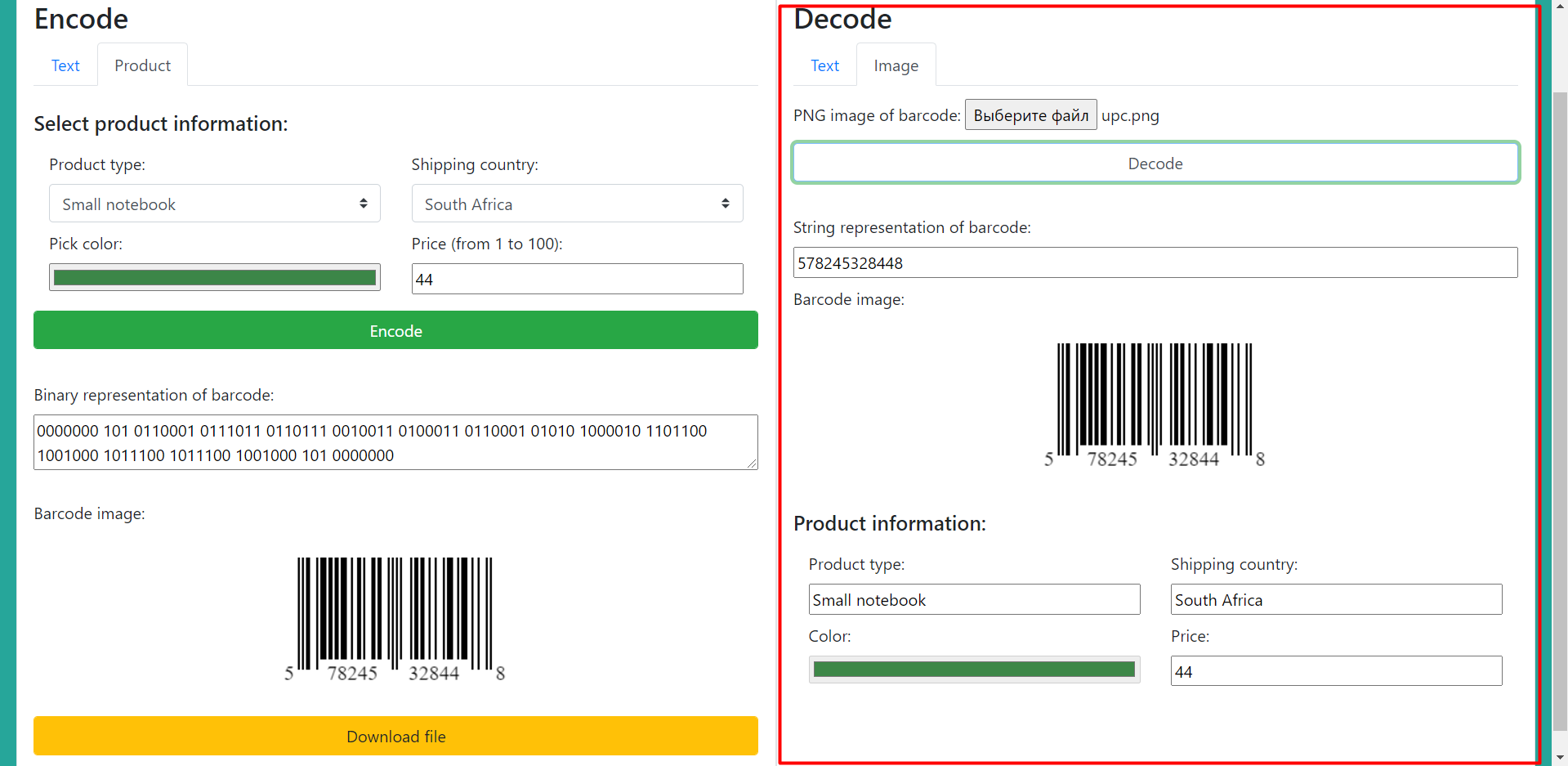
* Кодування вхідної інформації отриманої у вигляді параметрів продукту



* Декодування вхідної інформації отриманої штрих кодової позначки у вигляді бінарної строки



* Декодування вхідної інформації отриманої штрих кодової позначки у вигляді файлу із зображенням



**Висновок**

Під час вивчення дисципліни мною був виконаний комплекс лабораторних робіт. В результаті було створено програму, що включає в собі метод кодування вхідної інформації, метод друкування (зберігання у виді файлу) штрихової позначки, програму декодування файлу штрихової позначки та вивід результати в інтерфейсі користувача.