Министерство образования Республики Беларусь

Учреждение образования

БелорусскиЙ государственный университет

информатики и радиоэлектроники

Факультет компьютерных систем и сетей

Кафедра программного обеспечения информационных технологий

# оТЧЕТ

по лабораторной работе

на тему:

МЕТРИКИ ХОЛСТЕДА

Выполнили: Булдыго Д.Ю.

Павловец С.В.

Проверил: Acc. С.В. Болтак

Минск, 2021

Основу метрик Холстеда составляют *шесть базовых метрик* программы:

* 1 *–* словарь операторов (число уникальных операторов программы);
* 2 *–* словарь операндов (число уникальных операндов программы);
* *N1 –* общее число операторов в программе;
* *N1 –* общее число операндов в программе;
* *f1j –* число вхождений *j*-го оператора, *j* = 1, 2, …, ɳ 1;
* *f2i –* число вхождений *i*-го операнда, *i* = 1, 2, …, ɳ 2.

Также выделяют три производные метрики:

* *N = N1 + N2* – длина программы
*  = 1 + 2 – словарь программы
* *V = N log2*– объем программы

**Фрагмент программы работы с динамическими структурами**

**данных на языке TypeScript**

// Method to push data to stack

Stack.prototype.push = function(data: any){

if (!this.empty()){

// Create new node and make it top element

let node = new ListNode(data);

node.next = this.top;

this.top = node;

}

else

this.top = new ListNode(data);

}

// Check whether tree is empty

BinarySearchTree.prototype.empty = function(): boolean{

return this.root === null;

}

try{

// Initialize stack, queue, tree

let stack = new Stack(), queue = new Queue(), tree = new BinarySearchTree();

// Initialize data for structures

let keys: number[] = [], samples: string[] = ['cherry', 'apple', 'pear', 'strawberry', 'grape', 'juniper'];

// Filling structures and outputting array

for(let i in samples){

stack.push(samples[i]);

queue.enqueue(samples[i]);

keys[i] = Math.floor(Math.random() \* 90 + 10);

tree.insert(keys[i], samples[i]);

console.log(keys[i], samples[i]);

}

// Releasing stack

console.log('Stack: ')

console.log('---------------------------------');

while (!stack.empty()){

console.log(stack.peek());

stack.pop();

}

console.log('---------------------------------');

try{

}

catch{

// Test catch

const break\_1: number = 0o0000;

do{

for(let j = 1; j < 10; j++){

--j;

++j;

}

} while(break\_1);

}

Таблица 1. Расчет базовых метрик Холстеда

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| j | Оператор | F1j |  | i | Операнд | F2i |
| 1 | . | 23 |  | 1 | Stack | 1 |
| 2 | = | 14 |  | 2 | prototype | 2 |
| 3 | {…} | 6 |  | 3 | push | 1 |
| 4 | if(…)else | 1 |  | 4 | this | 5 |
| 5 | (…) | 1 |  | 5 | node | 3 |
| 6 | ! | 2 |  | 6 | data | 2 |
| 7 | empty(…) | 2 |  | 7 | next | 1 |
| 8 | new | 5 |  | 8 | top | 3 |
| 9 | ListNode(…) | 2 |  | 9 | BinarySearchTree | 1 |
| 10 | ; | 22 |  | 10 | empty | 1 |
| 11 | return | 1 |  | 11 | root | 1 |
| 12 | === | 1 |  | 12 | null | 1 |
| 13 | try/catch/finally | 1 |  | 13 | stack | 5 |
| 14 | Stack(…) | 1 |  | 14 | queue | 2 |
| 15 | , | 10 |  | 15 | tree | 2 |
| 16 | Queue(…) | 1 |  | 16 | keys | 4 |
| 17 | BinarySearchTree  (…) | 1 |  | 17 | samples | 6 |
| 18 | : | 3 |  | 18 | ‘cherry’ | 1 |
| 19 | […] | 9 |  | 19 | i | 8 |
| 20 | for/in(…) | 1 |  | 20 | Math | 2 |
| 21 | push(…) | 1 |  | 21 | 90 | 1 |
| 22 | enqueue(…) | 1 |  | 22 | 10 | 2 |
| 23 | floor(…) | 1 |  | 23 | console | 5 |
| 24 | random(…) | 1 |  | 24 | ‘Stack:’ | 1 |
| 25 | \* | 1 |  | 25 | ‘----------------------------------’ | 2 |
| 26 | + | 1 |  | 26 | break\_1 | 2 |
| 27 | insert(…) | 1 |  | 27 | 0o0000 | 1 |
| 28 | log(…) | 5 |  | 28 | j | 5 |
| 29 | while(…) | 1 |  | 29 | 1 | 1 |
| 30 | peek(…) | 1 |  |  |  |  |
| 31 | pop(…) | 1 |  |  |  |  |
| 32 | do{…}while(…) | 1 |  |  |  |  |
| 33 | for(…;…;…) | 1 |  |  |  |  |
| 34 | < | 1 |  |  |  |  |
| 35 | ++ | 2 |  |  |  |  |
| 36 | -- | 1 |  |  |  |  |

Таблица 2. Базовые метрики

|  |  |
| --- | --- |
| Метрика | Значение |
| Cловарь операторов | 36 |
| Cловарь операндов | 29 |
| Общее число операторов | 128 |
| Общее число операндов | 72 |
| Число вхождений *j*-го оператора | Значения приведены в таблице 1 |
| Число вхождений *i*-го операнда | Значения приведены в таблице 1 |

Таблица 3. Производные метрики программы

|  |  |
| --- | --- |
| Метрика | Значение |
| Словарь программы | 65 |
| Длина программы | 200 |
| Объем программы | 1204 |

**Результат работы программы**

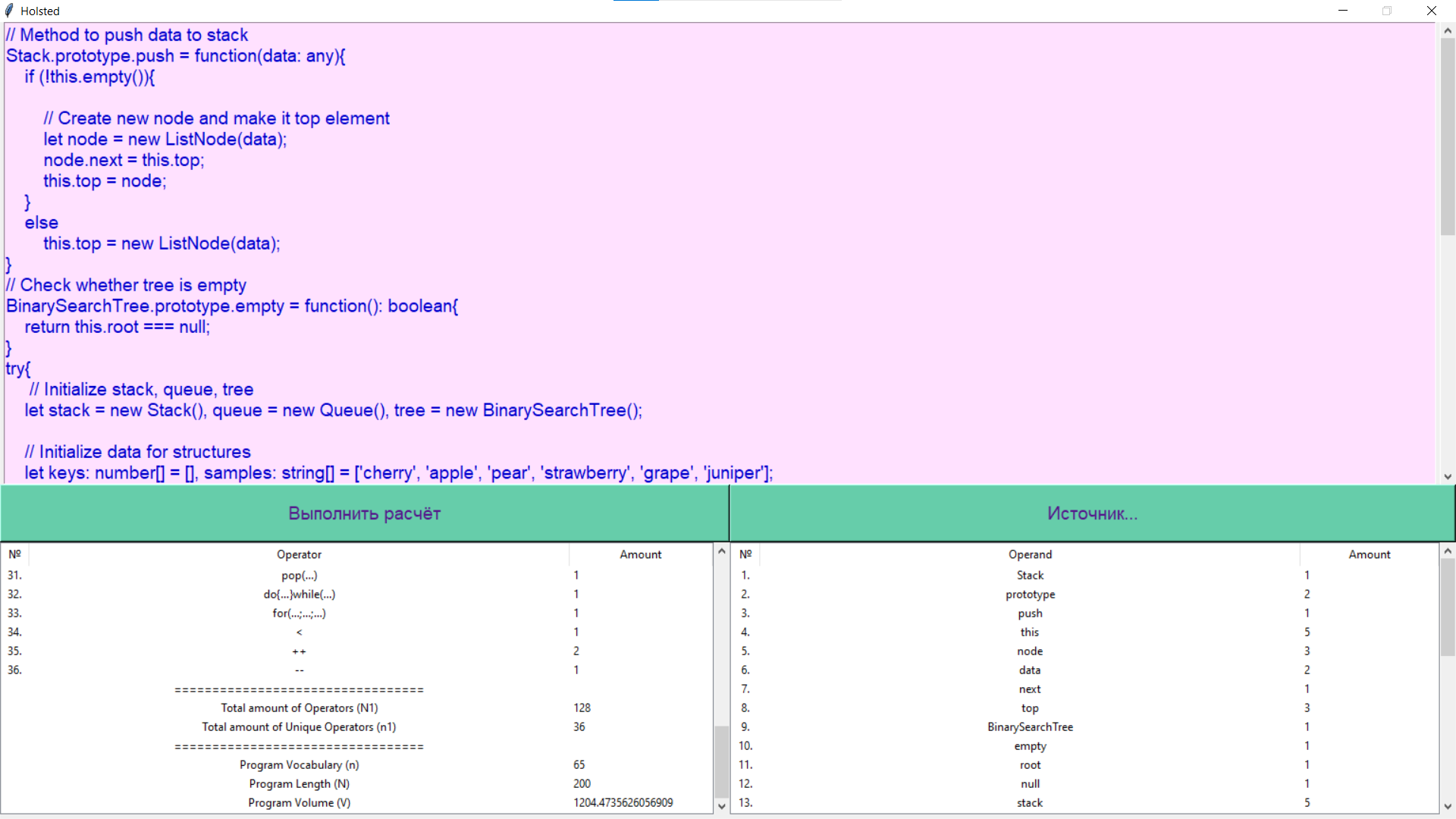


Рисунок 1 – Результат работы программы