|  |  |  |
| --- | --- | --- |
| МИНОБРНАУКИ РОССИИ | | |
| Федеральное государственное автономное  образовательное учреждение высшего образования  «Пермский государственный национальный  исследовательский университет» | | |
|  | Институт компьютерных наук и технологий | |
| **ОТЧЁТ**  по индивидуальной работе №2  по дисциплине «Языки программирования»  Вариант 4 | | |
|  | | Работу выполнил  студент группы ПМИ-9,10-2023 1 курса  Румянцев В.И.  «16» июня 2024 г. |
| Работу проверил  \_\_\_\_\_\_\_\_\_\_\_\_\_\_ Фамилия И.О.  «\_\_\_» \_\_\_\_\_\_\_\_\_ 2024 г. |
| Пермь 2024 | | |

СОДЕРЖАНИЕ

[Постановка задачи 3](#_Toc169483386)

[Алгоритм решения 5](#_Toc169483387)

[Тестирование 6](#_Toc169483388)

[Код программы 10](#_Toc169483389)

# Постановка задачи

Абстрактная вычислительная машина имеет один регистр и шесть инструкций:

LD A — Помещает операнд А в регистр

ST A — Помещает содержимое регистра в переменную с именем А

AD A — Прибавляет значение А к регистру. Результат остается в регистре.

SB A — Вычитает значение А из регистра. Результат остается в регистре.

ML A — Умножает содержимое регистра на значение переменной с именем A. Результат остается в регистре.

DV A — Делит содержимое регистра на значение А. Результат остается в регистре.

Дано выражение в постфиксной форме, состоящее из однобуквенных операндов и операций «+», «–», «\*», «/». Написать программу вывода инструкций вычислительной машины, необходимых для вычисления выражения. Результат вычисления должен оставаться в регистре, разрешается использовать обозначения Tn для временных переменных.

Пример: для выражения ABC\*+DE-/ перечень инструкций имеет вид

LD B

ML C

ST T1

LD A

AD T1

ST T1

LD D

SB E

ST T2

LD T1

DV T2

Замечание: Постфиксной формой записи выражения a b называется запись, в которой знак операции размещен за операндами a b.

Примеры:

a - b → a b –

a \* b + c → a b \* c +

a \* (b + c) → a b c + \*

a + b / c / d \* e → a b c / d / e \* +

# Алгоритм решения

1. Создал динамическую структуру данных – односвязный список (стек) для хранения переменных.

2. Инициализировал переменную `n` для подсчёта номера следующей временной переменной `Tn`.

3. Инициализировал пустой список `free\_names` для хранения освободившихся имён временных переменных `Tn`.

4. Написал функцию `get\_name\_of\_temp\_var`, которая определяет имена для временных переменных `Tn` и добавляет освободившиеся имена в список `free\_names`.

5. Написал функцию `push`, которая добавляет новый элемент в стек или выполняет операции над переменными из стека, если необходимо.

6. Написал функцию `main`, которая принимает на вход выражение в постфиксной форме и вызывает функцию `push` для каждого элемента выражения.

7. В функции `main` написал проверку на то, что в стеке остался только один элемент после выполнения всех операций. Если в стеке осталось больше одного элемента или не осталось ни одного, то выводится сообщение об ошибке.

8. В функции `main` написал вывод результата в виде списка инструкций вычислительной машины.

9. Написал код для ввода выражения от пользователя и вызова функции `main`.

10. Добавил комментарии к коду для лучшего понимания работы программы.

11. Проверил работу программы на различных входных данных.

# Тестирование

1. Запрос: «ABC\*+DE-/»

Ответ: «

Результат:

LD B

ML C

ST T1

LD A

AD T1

ST T1

LD D

SB E

ST T2

LD T1

DV T2

»

1. Запрос: «AB+CD+-EF\*GH/+\*»

Ответ: «

Результат:

LD A

AD B

ST T1

LD C

AD D

ST T2

LD T1

SB T2

ST T1

LD E

ML F

ST T2

LD G

DV H

ST T3

LD T2

AD T3

ST T2

LD T1

ML T2

»

1. Запрос: «abc/d/e\*+»

Ответ: «

Результат:

LD b

DV c

ST T1

LD T1

DV d

ST T1

LD T1

ML e

ST T1

LD a

AD T1

»

1. Запрос: «»

Ответ: «

Результат отсутствует!

Пожалуйста, перепроверьте корректность вашего выражения и запросите новый результат.

»

1. Запрос: «ABC-»

Ответ: «

Программа окончила работу с лишним(и) операндом(ами) в регистре: "A".

Это связано с излишним их количеством на входе (например, было подано выражение "ABC+").

Совет: В случае, если вы не уверены в корректности вашего запроса - перепроверьте выражение, которое было подано программе на вход и запросите новый результат!

Результат:

LD B

SB C

»

1. Запрос: «A/»

Ответ: «

Во время работы программы произошла ошибка из-за нехватки переменных для работы.

Такая ошибка происходит, например, когда на вход программе подано такое выражение: "A+", вместо такого "AB+".

Перепроверьте ваше выражение и введите его корректно.

»

1. Запрос: «JVSNVFKS»

Ответ: «

Программа окончила работу с лишним(и) операндом(ами) в регистре: "K", "F", "V", "N", "S", "V", "J".

Это связано с излишним их количеством на входе (например, было подано выражение "ABC+").

Совет: В случае, если вы не уверены в корректности вашего запроса - перепроверьте выражение, которое было подано программе на вход и запросите новый результат!

»

1. Запрос: «АППААП++++++»

Ответ: «

Во время работы программы произошла ошибка из-за нехватки переменных для работы.

Такая ошибка происходит, например, когда на вход программе подано такое выражение: "A+", вместо такого "AB+".

Перепроверьте ваше выражение и введите его корректно.

»

1. Запрос: «Абапре+++++»

Ответ: «

Результат:

LD р

AD е

ST T1

LD п

AD T1

ST T1

LD а

AD T1

ST T1

LD б

AD T1

ST T1

LD А

AD T1»

# Код программы

# Вариант 4. Румянцев Владимир ПМИ-10.

class Node:

def \_\_init\_\_(self, data=None):

self.data = data

self.next = None

class Stack:

def \_\_init\_\_(self):

self.head = None

self.n = 1 # Число n для временных переменных Tn

self.free\_names = [] # Пополняется освободившимися Tn

def get\_name\_of\_temp\_var(self, var1, var2): # Функция получения имени для временной переменной Tn

if (len(var1) > 1) and (var1[0] == 'T'): # Если первый операнд в формате Tn, ...

try:

a = int(var1[1:])

if a < self.n:

self.free\_names.append(var1) # ... то пополнить им список освободившихся Tn

except ValueError:

pass

if (len(var2) > 1) and (var2[0] == 'T'): # Если второй операнд в формате Tn, ...

try:

a = int(var2[1:])

if a < self.n:

self.free\_names.append(var2) # ... то пополнить им список освободившихся Tn

except ValueError:

pass

if self.free\_names: # Если в списке освободившихся Tn что-то есть,

return self.free\_names.pop(0) # то вернуть первый элемент

temp\_var = f'T{self.n}' # Если освободившихся имён нет, то создать и вернуть новое имя,

self.n += 1 # исходя из "self.n". Так-же увеличить "self.n" на 1

return temp\_var

def push(self, new\_data=None): # Добавление в стек нового элемента

if not (new\_data in '\*/+-'): # Если это операнд, то просто добавить его в стек

new\_node = Node(new\_data)

new\_node.next = self.head

self.head = new\_node

return '' # Возвращаем пустую строку

else: # Если это операция...

if new\_data == '+':

operation = 'AD'

elif new\_data == '-':

operation = 'SB'

elif new\_data == '\*':

operation = 'ML'

else:

operation = 'DV'

try:

variable\_1 = self.head.next.data # Получить данные первого и второго операндов

variable\_2 = self.head.data

except AttributeError: # В стеке не хватает операндов, вызвать ошибку и окончить работу

raise Exception(f'Во время работы программы произошла ошибка из-за нехватки переменных для работы.\n'

f'Такая ошибка происходит, например, когда на вход программе подано такое выражение: '

f'"A+", вместо такого "AB+".\nПерепроверьте ваше выражение и введите его корректно.')

temp\_var = self.get\_name\_of\_temp\_var(variable\_1, variable\_2) # Получить имя для временной переменной

line = f'LD {variable\_1}\n{operation} {variable\_2}\nST {temp\_var}\n' # Перечень инструкций

new\_node = Node(temp\_var) # Помещение временной переменной в стек

new\_node.next = self.head.next.next # Переопределение ссылок для удаления операндов

self.head = new\_node

return line # Возвращаем перечень инструкций

def main(input\_data): # Главная функция, на входе получает выражение в постфиксной форме

stack = Stack()

line = ''

for i in input\_data: # Проходимся по каждому элементу в строке входных данных

line += stack.push(i) # В line дописываем результат функции push (конкретные инструкции)

res\_line = '' # Создание переменной для итогового ответа пользователю

try: # Проверка на то, остались ли элементы в стеке,

current = stack.head.next # в норме останется только один элемент

if current: # Если элементов в стеке осталось больше одного, ...

res\_line = f'\nПрограмма окончила работу с лишним(и) операндом(ами) в регистре: '

while current: # ... то сообщим об этом в итоговом ответе

res\_line += f'"{current.data}", '

current = current.next

res\_line = (f'{res\_line[:-2]}.\n'

f'Это связано с излишним их количеством на входе (например, было подано выражение "ABC+").\n'

f'Совет: В случае, если вы не уверены в корректности вашего запроса - перепроверьте выражение, '

f'которое было подано программе на вход и запросите новый результат!\n')

except AttributeError: # Ошибка произошла из-за отсутствия в стеке даже одного элемента,

pass # Значит пользователь дал на вход пустую стоку

for i in range(len(line) - 2, -1, -1): # Т.к. согласно условию результат вычислений остаётся в регистре,

if (line[i] == 'S') and (line[i + 1] == 'T'): # нужно избавиться от последней инструкции ST

res\_line += f'\nРезультат:\n{line[:i].strip()}' # В переменную с конечным ответом записываем результат

break

if not res\_line: # Результат отсутствует, т.к. на вход пришла пустая строка

print('Результат отсутствует!\n'

'Пожалуйста, перепроверьте корректность вашего выражения и запросите новый результат.')

else:

print(res\_line) # Сообщить пользователю итог (перечень инструкций)

if \_\_name\_\_ == '\_\_main\_\_':

user\_input = input('\nВведите выражение в постфиксной форме (пример: ABC\*+): ') # Ввод пользователя

while True:

try:

main(user\_input) # Начало работы с данными от пользователя

except Exception as ex: # В случае ошибки, о ней сообщаем пользователю

print(ex)

finally: # Вне зависимости от результата, просим новое выражение

user\_input = input('\n----------------------------------------------------------------'

'----------------------------------------------------------------\n'

'\nВведите новое выражение: ')