

СПЕЦІАЛЬНІ РОЗДІЛИ ОБЧИСЛЮВАЛЬНОЇ МАТЕМАТИКИ

Комп'ютерний практикум №2

Багаторозрядна модулярна арифметика

ФБ-23 Моїсеєнко Дмитро

Мета роботи:

Отримання практичних навичок програмної реалізації багаторозрядної арифметики; ознайомлення з прийомами ефективної реалізації критичних по часу ділянок програмного коду та методами оцінки їх ефективності.

Завдання до комп'ютерного практикуму:

А) Доопрацювати бібліотеку для роботи з m -бітними цілими числами, створену на комп'ютерному практикумі №1, додавши до неї такі операції:

- 1) обчислення НСД та НСК двох чисел;
- 2) додавання чисел за модулем;
- 3) віднімання чисел за модулем;
- 4) множення чисел та піднесення чисел до квадрату за модулем;
- 5) піднесення числа до багаторозрядного степеня d по модулю n .

Модулярну арифметику рекомендовано реалізовувати на базі редукції Баррета, піднесення до степеня – на базі схеми Горнера.

Мова програмування, семантика функцій та спосіб реалізації можуть обиратись довільним чином.

Окрім основного завдання, ви також можете виконати додаткове завдання згідно варіанту.

Б) Проконтролювати коректність реалізації алгоритмів; зокрема, для декількох багаторозрядних a, b, c, n перевірити тотожності

В) Обчислити середній час виконання реалізованих арифметичних операцій. Підрахувати кількість тактів процесора (або інших одиниць виміру часу) на кожну операцію. Результати подати у вигляді таблиць або діаграм.

Хід роботи:

Так само як із попереднього першого лабораторної роботи бібліотеку `bighum`

Результати виконання роботи:

```
1 0 & n
2 1312228813392838887157888878891724617634748885224127947872441398288848863758488124588818332428151326674811232988848899133781342878827142678443221843828819258828811378788655884886478188111424
331828711715888291623588891141285454251317834888543582924758488678844738844826345788857881438898889787852728841886721637583887781787167328388417888481137818154374246547717438323782882635826618888814274
816298472727273352737718738158888822321858884824747183978838282635873771338422279953219331778864884875741218428884342813835887467792811885882297218273135557332811287934462461186321475388419848888393
39432613818
4 4) Abn + Abn
5 False
6 5) Abn + Abn
7 True
8 6) Abn.rshlft
9 -bound method bn.rshlft of 314666120158388785, 1516771417881784653, 13623288857478937710, 15541788241778227158, 1816682327517787468, 11349885557178829365, 17542262653952221132, 2448155623478477238, 6879
13831488788634, 8767891597864622139, 18752431878843225141, 18968888538885924558, 9943885367822885432, 686923828478517218, 3863847218374341339, 6752884643784488892, 14678872885177181847, 13826478482888758728, 4588
85994631272231, 7884258878442938185, 11983277891714537558, 12365923788728378185, 15551648155862588488, 134488888889781338397, 13838166191213775389, 16587318885829856378, 1413388536349888148, 631935884248183859, 1
348393539238819888, 1373123363495215588, 695899729786289488, 7788893218888935788
10 7) Abn.rshlft(0)
11 8) Abn.base(10)
Traceback (most recent call last)
<ipython-input-8-95738848843> in <module>
----> 1 Abn.base(10)
<ipython>: 'int' object is not callable
12 9) Abn.base(8)
13 1178188136371858888878515258488277612182582111788478787523886388127888165655884888588688749393859018494484555138314515856733894788834871648858878888841989939838747847719646558613866161587385
7832428146888894914419885249486445175013552448888558855287363638234488885885588874471218388725287384521581295829158888887324358885928843478274971632928888465858858834228378574117988888848888736648478
6882419534474928413848823888284232888884977269125487821677441848862942816822985626888937829384647821337839863771588994883888221911389925158793465788845343483881597844518185188825884618921192841938
```

```
dmity@dmity-virtual-machine:~/Lab1$ python3 Test.py
[*] Checking the correctness of the conversion...
A == Abn: True
A16 == Abn16: True
A2 == Abn2: True
[!] Conversion to common bases seems right
[*] Checking addition...
Abn + Bbn == Bbn + Abn: True
A + B == Abn + Bbn: True
(A + B) + C == Abn + (Bbn + Cbn): True
D + A == Dbn + Abn, where D is negative number: True
[!] Addition seems right checking subtraction...
Abn - Bbn == Bbn - Abn: True
A - B == Abn - Bbn: True
(A - B) - C == Abn - (Bbn - Cbn): True
D - A == Dbn - Abn, where D is negative number: True
[!] Subtraction seems right
[*] Checking multiplication...
Abn * Bbn == Bbn * Abn: True
A * B == Abn * Bbn: True
(A * B) * C == Abn * (Bbn * Cbn): True
Abn * 123 = Abn+Abn...+Abn times 123: True
(Abn+Bbn)*Cbn == Abn*Cbn + Bbn*Cbn: True
[!] Multiplication seems right
```

Додавання

```
107 function calls in 0.000 seconds

Ordered by: standard name

ncalls  tottime  percall  cumtime  percall  filename:lineno(function)
1      0.000    0.000    0.000    0.000    <string>:1(<module>)
1      0.000    0.000    0.000    0.000    Test1.1.py:9(add)
1      0.000    0.000    0.000    0.000    bignum.py:12(__init__)
1      0.000    0.000    0.000    0.000    bignum.py:57(__add__)
1      0.000    0.000    0.000    0.000    conv_types.py:15(convert)
1      0.000    0.000    0.000    0.000    {built-in method builtins.exec}
2      0.000    0.000    0.000    0.000    {built-in method builtins.isinstance}
65     0.000    0.000    0.000    0.000    {built-in method builtins.len}
1      0.000    0.000    0.000    0.000    {built-in method builtins.max}
32     0.000    0.000    0.000    0.000    {method 'append' of 'list' objects}
1      0.000    0.000    0.000    0.000    {method 'disable' of '_lsprof.Profiler' objects}
```

Віднімання

108 function calls in 0.000 seconds					
Ordered by: standard name					
ncalls	tottime	percall	cumtime	percall	filename:lineno(function)
1	0.000	0.000	0.000	0.000	<string>:1(<module>)
1	0.000	0.000	0.000	0.000	Test1.1.py:11(sub)
1	0.000	0.000	0.000	0.000	bignum.py:12(__init__)
1	0.000	0.000	0.000	0.000	bignum.py:82(sub_s)
1	0.000	0.000	0.000	0.000	bignum.py:98(__sub__)
1	0.000	0.000	0.000	0.000	conv_types.py:15(convert)
1	0.000	0.000	0.000	0.000	{built-in method builtins.exec}
2	0.000	0.000	0.000	0.000	{built-in method builtins.isinstance}
65	0.000	0.000	0.000	0.000	{built-in method builtins.len}
1	0.000	0.000	0.000	0.000	{built-in method builtins.max}
32	0.000	0.000	0.000	0.000	{method 'append' of 'list' objects}
1	0.000	0.000	0.000	0.000	{method 'disable' of '_lsprof.Profiler' objects}
114485 function calls (111995 primitive calls) in 0.144 seconds					

Піднесення в степінь

5967210 function calls in 10.189 seconds					
Ordered by: standard name					
ncalls	totttime	percall	cumtime	percall	filename:lineno(function)
1	0.000	0.000	10.189	10.189	<string>:1(<module>)
1	0.000	0.000	10.189	10.189	Test1.1.py:15(div)
3923	0.025	0.000	0.331	0.000	bignum.py:12(__init__)
1	0.000	0.000	0.000	0.000	bignum.py:139(__lt__)
1924	0.041	0.000	0.059	0.000	bignum.py:146(__eq__)
1924	0.007	0.000	0.007	0.000	bignum.py:147(<listcomp>)
1924	0.001	0.000	0.001	0.000	bignum.py:148(<listcomp>)
1	0.049	0.049	10.189	10.189	bignum.py:225(divMod)
1	0.000	0.000	10.189	10.189	bignum.py:254(__truediv__)
1921	0.003	0.000	0.930	0.000	bignum.py:262(lshift)
3843	0.008	0.000	2.679	0.001	bignum.py:266(rshift)
7686	1.716	0.000	1.716	0.000	bignum.py:28(base10)
1921	0.926	0.000	0.927	0.000	bignum.py:316(lshiftBits)
3843	2.668	0.001	2.671	0.001	bignum.py:334(rshiftBits)
998	0.077	0.000	0.105	0.000	bignum.py:57(__add__)
999	3.168	0.003	4.319	0.004	bignum.py:82(sub_s)
998	0.012	0.000	4.623	0.005	bignum.py:98(__sub__)
3923	0.298	0.000	0.305	0.000	conv_types.py:15(convert)
1925	0.003	0.000	0.003	0.000	conv_types.py:3(getDigits)
1	0.000	0.000	10.189	10.189	{built-in method builtins.exec}
5921	0.004	0.000	0.004	0.000	{built-in method builtins.isinstance}
3941618	0.791	0.000	0.791	0.000	{built-in method builtins.len}
1997	0.005	0.000	0.005	0.000	{built-in method builtins.max}
1950155	0.384	0.000	0.384	0.000	{method 'append' of 'list' objects}
1	0.000	0.000	0.000	0.000	{method 'disable' of '_lsprof.Profiler' objects}
29760	0.005	0.000	0.005	0.000	{method 'pop' of 'list' objects}

Множення

62183 function calls (60839 primitive calls) in 0.069 seconds					
Ordered by: standard name					
ncalls	totttime	percall	cumtime	percall	filename:lineno(function)
1	0.000	0.000	0.069	0.069	<string>:1(<module>)
1	0.000	0.000	0.069	0.069	Test1.1.py:17(pow)
6100	0.013	0.000	0.028	0.000	bignum.py:12(__init__)
3	0.000	0.000	0.000	0.000	bignum.py:146(__eq__)
3	0.000	0.000	0.000	0.000	bignum.py:147(<listcomp>)
3	0.000	0.000	0.000	0.000	bignum.py:148(<listcomp>)
457	0.001	0.000	0.001	0.000	bignum.py:158(mulStep)
681/9	0.003	0.000	0.069	0.008	bignum.py:171(__mul__)
1	0.000	0.000	0.069	0.069	bignum.py:199(__pow__)
681/9	0.013	0.000	0.069	0.008	bignum.py:270(karatsubaStep)
1568	0.002	0.000	0.002	0.000	bignum.py:28(base10)
905	0.001	0.000	0.001	0.000	bignum.py:310(shiftLeft)
1	0.000	0.000	0.000	0.000	bignum.py:34(baseN)
1801	0.012	0.000	0.025	0.000	bignum.py:57(__add__)
448	0.003	0.000	0.004	0.000	bignum.py:82(sub_s)
448	0.001	0.000	0.006	0.000	bignum.py:98(__sub__)
6100	0.009	0.000	0.014	0.000	conv_types.py:15(convert)
686	0.000	0.000	0.000	0.000	conv_types.py:3(getDigits)
1	0.000	0.000	0.069	0.069	{built-in method builtins.exec}
11514	0.004	0.000	0.004	0.000	{built-in method builtins.isinstance}
19835	0.005	0.000	0.005	0.000	{built-in method builtins.len}
3154	0.002	0.000	0.002	0.000	{built-in method builtins.max}
1	0.000	0.000	0.000	0.000	{built-in method math.log}
7787	0.001	0.000	0.001	0.000	{method 'append' of 'list' objects}
1	0.000	0.000	0.000	0.000	{method 'disable' of '_lsprof.Profiler' objects}
1	0.000	0.000	0.000	0.000	{method 'find' of 'str' objects}
1	0.000	0.000	0.000	0.000	{method 'lstrip' of 'str' objects}