OS 2022fall 11.10 hw9

20307140008 施君豪

运行环境：Windows 10 (pycharm)

1. **第十七章**
2. **运行结果详见下列代码**

(base) C:\Users\Michael Shi\Desktop\ostep-homework-dev-zhongyl\vm-freespace>malloc.py -n 10 -H 0 -p BEST -s 0

seed 0

size 100

baseAddr 1000

headerSize 0

alignment -1

policy BEST

listOrder ADDRSORT

coalesce False

numOps 10

range 10

percentAlloc 50

allocList

compute False

ptr[0] = Alloc(3) returned ?

List?

Free(ptr[0])

returned ?

List?

ptr[1] = Alloc(5) returned ?

List?

Free(ptr[1])

returned ?

List?

ptr[2] = Alloc(8) returned ?

List?

Free(ptr[2])

returned ?

List?

ptr[3] = Alloc(8) returned ?

List?

Free(ptr[3])

returned ?

List?

ptr[4] = Alloc(2) returned ?

List?

ptr[5] = Alloc(7) returned ?

List?

**实际结果详见下方，可以观察到中间空闲的碎片越来越多。**

**ptr[0] = Alloc(3) returned 1000**

**Free List [ Size 1 ]: [ addr:1003 sz:97 ]**

**Free(ptr[0])**

**returned 0**

**Free List [ Size 2 ]: [ addr:1000 sz:3 ][ addr:1003 sz:97 ]**

**ptr[1] = Alloc(5) returned 1003**

**Free List [ Size 2 ]: [ addr:1000 sz:3 ][ addr:1008 sz:92 ]**

**Free(ptr[1])**

**returned 0**

**Free List [ Size 3 ]: [ addr:1000 sz:3 ][ addr:1003 sz:5 ][ addr:1008 sz:92 ]**

**ptr[2] = Alloc(8) returned 1008**

**Free List [ Size 3 ]: [ addr:1000 sz:3 ][ addr:1003 sz:5 ][ addr:1016 sz:84 ]**

**Free(ptr[2])**

**returned 0**

**Free List [ Size 4 ]: [ addr:1000 sz:3 ][ addr:1003 sz:5 ][ addr:1008 sz:8 ][ addr:1016 sz:84 ]**

**ptr[3] = Alloc(8) returned 1008**

**Free List [ Size 3 ]: [ addr:1000 sz:3 ][ addr:1003 sz:5 ][ addr:1016 sz:84 ]**

**Free(ptr[3])**

**returned 0**

**Free List [ Size 4 ]: [ addr:1000 sz:3 ][ addr:1003 sz:5 ][ addr:1008 sz:8 ][ addr:1016 sz:84 ]**

**ptr[4] = Alloc(2) returned 1000**

**Free List [ Size 4 ]: [ addr:1002 sz:1 ][ addr:1003 sz:5 ][ addr:1008 sz:8 ][ addr:1016 sz:84 ]**

**ptr[5] = Alloc(7) returned 1008**

**Free List [ Size 4 ]: [ addr:1002 sz:1 ][ addr:1003 sz:5 ][ addr:1015 sz:1 ][ addr:1016 sz:84 ]**

1. **运行结果详见下列代码，空闲碎片变得更多了**

(base) C:\Users\Michael Shi\Desktop\ostep-homework-dev-zhongyl\vm-freespace>malloc.py -n 10 -H 0 -p WORST -s 0 -c

seed 0

size 100

baseAddr 1000

headerSize 0

alignment -1

policy WORST

listOrder ADDRSORT

coalesce False

numOps 10

range 10

percentAlloc 50

allocList

compute True

ptr[0] = Alloc(3) returned 1000 (searched 1 elements)

Free List [ Size 1 ]: [ addr:1003 sz:97 ]

Free(ptr[0])

returned 0

Free List [ Size 2 ]: [ addr:1000 sz:3 ][ addr:1003 sz:97 ]

ptr[1] = Alloc(5) returned 1003 (searched 2 elements)

Free List [ Size 2 ]: [ addr:1000 sz:3 ][ addr:1008 sz:92 ]

Free(ptr[1])

returned 0

Free List [ Size 3 ]: [ addr:1000 sz:3 ][ addr:1003 sz:5 ][ addr:1008 sz:92 ]

ptr[2] = Alloc(8) returned 1008 (searched 3 elements)

Free List [ Size 3 ]: [ addr:1000 sz:3 ][ addr:1003 sz:5 ][ addr:1016 sz:84 ]

Free(ptr[2])

returned 0

Free List [ Size 4 ]: [ addr:1000 sz:3 ][ addr:1003 sz:5 ][ addr:1008 sz:8 ][ addr:1016 sz:84 ]

ptr[3] = Alloc(8) returned 1016 (searched 4 elements)

Free List [ Size 4 ]: [ addr:1000 sz:3 ][ addr:1003 sz:5 ][ addr:1008 sz:8 ][ addr:1024 sz:76 ]

Free(ptr[3])

returned 0

Free List [ Size 5 ]: [ addr:1000 sz:3 ][ addr:1003 sz:5 ][ addr:1008 sz:8 ][ addr:1016 sz:8 ][ addr:1024 sz:76 ]

ptr[4] = Alloc(2) returned 1024 (searched 5 elements)

Free List [ Size 5 ]: [ addr:1000 sz:3 ][ addr:1003 sz:5 ][ addr:1008 sz:8 ][ addr:1016 sz:8 ][ addr:1026 sz:74 ]

ptr[5] = Alloc(7) returned 1026 (searched 5 elements)

Free List [ Size 5 ]: [ addr:1000 sz:3 ][ addr:1003 sz:5 ][ addr:1008 sz:8 ][ addr:1016 sz:8 ][ addr:1033 sz:67 ]

1. **运行结果详见下列代码，空闲碎片相较于WORST更少，最终空闲碎片和BEST相同，且运行速度比前两者更快。**

(base) C:\Users\Michael Shi\Desktop\ostep-homework-dev-zhongyl\vm-freespace>malloc.py -n 10 -H 0 -p FIRST -s 0 -c

seed 0

size 100

baseAddr 1000

headerSize 0

alignment -1

policy FIRST

listOrder ADDRSORT

coalesce False

numOps 10

range 10

percentAlloc 50

allocList

compute True

ptr[0] = Alloc(3) returned 1000 (searched 1 elements)

Free List [ Size 1 ]: [ addr:1003 sz:97 ]

Free(ptr[0])

returned 0

Free List [ Size 2 ]: [ addr:1000 sz:3 ][ addr:1003 sz:97 ]

ptr[1] = Alloc(5) returned 1003 (searched 2 elements)

Free List [ Size 2 ]: [ addr:1000 sz:3 ][ addr:1008 sz:92 ]

Free(ptr[1])

returned 0

Free List [ Size 3 ]: [ addr:1000 sz:3 ][ addr:1003 sz:5 ][ addr:1008 sz:92 ]

ptr[2] = Alloc(8) returned 1008 (searched 3 elements)

Free List [ Size 3 ]: [ addr:1000 sz:3 ][ addr:1003 sz:5 ][ addr:1016 sz:84 ]

Free(ptr[2])

returned 0

Free List [ Size 4 ]: [ addr:1000 sz:3 ][ addr:1003 sz:5 ][ addr:1008 sz:8 ][ addr:1016 sz:84 ]

ptr[3] = Alloc(8) returned 1008 (searched 3 elements)

Free List [ Size 3 ]: [ addr:1000 sz:3 ][ addr:1003 sz:5 ][ addr:1016 sz:84 ]

Free(ptr[3])

returned 0

Free List [ Size 4 ]: [ addr:1000 sz:3 ][ addr:1003 sz:5 ][ addr:1008 sz:8 ][ addr:1016 sz:84 ]

ptr[4] = Alloc(2) returned 1000 (searched 1 elements)

Free List [ Size 4 ]: [ addr:1002 sz:1 ][ addr:1003 sz:5 ][ addr:1008 sz:8 ][ addr:1016 sz:84 ]

ptr[5] = Alloc(7) returned 1008 (searched 3 elements)

Free List [ Size 4 ]: [ addr:1002 sz:1 ][ addr:1003 sz:5 ][ addr:1015 sz:1 ][ addr:1016 sz:84 ]

1. **对于BEST，WORST，以及FIRST而言，由于释放内存，插入空闲块时需要进行排序，故在free的时候速度均会放慢。**

**对于ADDRSORT，实际运行代码与上面实际结果没有区别，故不再展示。**

**对于SIZESORT+，最大块在最后，可以使FIRST算法变快，WORST算法变慢(遍历)**

**对于SIZESORT-，最小块在最后，可以使WORST算法变快，FIRST算法变慢(遍历)**

**对于BEST算法均无影响。**

**测试代码：**

**malloc.py -n 10 -H 0 -p BEST -s 0 -l ADDRSORT**

**malloc.py -n 10 -H 0 -p WORST -s 0 -l ADDRSORT**

**malloc.py -n 10 -H 0 -p FIRST -s 0 -l ADDRSORT**

**malloc.py -n 10 -H 0 -p BEST -s 0 -l SIZESORT+**

**malloc.py -n 10 -H 0 -p BEST -s 0 -l SIZESORT-**

**malloc.py -n 10 -H 0 -p WORST -s 0 -l SIZESORT+**

ptr[0] = Alloc(3) returned 1000 (searched 1 elements)

Free List [ Size 1 ]: [ addr:1003 sz:97 ]

Free(ptr[0])

returned 0

Free List [ Size 2 ]: [ addr:1000 sz:3 ][ addr:1003 sz:97 ]

ptr[1] = Alloc(5) returned 1003 (searched 2 elements)

Free List [ Size 2 ]: [ addr:1000 sz:3 ][ addr:1008 sz:92 ]

Free(ptr[1])

returned 0

Free List [ Size 3 ]: [ addr:1000 sz:3 ][ addr:1003 sz:5 ][ addr:1008 sz:92 ]

ptr[2] = Alloc(8) returned 1008 (searched 3 elements)

Free List [ Size 3 ]: [ addr:1000 sz:3 ][ addr:1003 sz:5 ][ addr:1016 sz:84 ]

Free(ptr[2])

returned 0

Free List [ Size 4 ]: [ addr:1000 sz:3 ][ addr:1003 sz:5 ][ addr:1008 sz:8 ][ addr:1016 sz:84 ]

ptr[3] = Alloc(8) returned 1016 (searched 4 elements)

Free List [ Size 4 ]: [ addr:1000 sz:3 ][ addr:1003 sz:5 ][ addr:1008 sz:8 ][ addr:1024 sz:76 ]

Free(ptr[3])

returned 0

Free List [ Size 5 ]: [ addr:1000 sz:3 ][ addr:1003 sz:5 ][ addr:1008 sz:8 ][ addr:1016 sz:8 ][ addr:1024 sz:76 ]

ptr[4] = Alloc(2) returned 1024 (searched 5 elements)

Free List [ Size 5 ]: [ addr:1000 sz:3 ][ addr:1003 sz:5 ][ addr:1008 sz:8 ][ addr:1016 sz:8 ][ addr:1026 sz:74 ]

ptr[5] = Alloc(7) returned 1026 (searched 5 elements)

Free List [ Size 5 ]: [ addr:1000 sz:3 ][ addr:1003 sz:5 ][ addr:1008 sz:8 ][ addr:1016 sz:8 ][ addr:1033 sz:67 ]

**malloc.py -n 10 -H 0 -p WORST -s 0 -l SIZESORT-**

ptr[0] = Alloc(3) returned 1000 (searched 1 elements)

Free List [ Size 1 ]: [ addr:1003 sz:97 ]

Free(ptr[0])

returned 0

Free List [ Size 2 ]: [ addr:1003 sz:97 ][ addr:1000 sz:3 ]

ptr[1] = Alloc(5) returned 1003 (searched 2 elements)

Free List [ Size 2 ]: [ addr:1008 sz:92 ][ addr:1000 sz:3 ]

Free(ptr[1])

returned 0

Free List [ Size 3 ]: [ addr:1008 sz:92 ][ addr:1003 sz:5 ][ addr:1000 sz:3 ]

ptr[2] = Alloc(8) returned 1008 (searched 3 elements)

Free List [ Size 3 ]: [ addr:1016 sz:84 ][ addr:1003 sz:5 ][ addr:1000 sz:3 ]

Free(ptr[2])

returned 0

Free List [ Size 4 ]: [ addr:1016 sz:84 ][ addr:1008 sz:8 ][ addr:1003 sz:5 ][ addr:1000 sz:3 ]

ptr[3] = Alloc(8) returned 1016 (searched 4 elements)

Free List [ Size 4 ]: [ addr:1024 sz:76 ][ addr:1008 sz:8 ][ addr:1003 sz:5 ][ addr:1000 sz:3 ]

Free(ptr[3])

returned 0

Free List [ Size 5 ]: [ addr:1024 sz:76 ][ addr:1008 sz:8 ][ addr:1016 sz:8 ][ addr:1003 sz:5 ][ addr:1000 sz:3 ]

ptr[4] = Alloc(2) returned 1024 (searched 5 elements)

Free List [ Size 5 ]: [ addr:1026 sz:74 ][ addr:1008 sz:8 ][ addr:1016 sz:8 ][ addr:1003 sz:5 ][ addr:1000 sz:3 ]

ptr[5] = Alloc(7) returned 1026 (searched 5 elements)

Free List [ Size 5 ]: [ addr:1033 sz:67 ][ addr:1008 sz:8 ][ addr:1016 sz:8 ][ addr:1003 sz:5 ][ addr:1000 sz:3 ]

**malloc.py -n 10 -H 0 -p FIRST -s 0 -l SIZESORT+**

ptr[0] = Alloc(3) returned 1000 (searched 1 elements)

Free List [ Size 1 ]: [ addr:1003 sz:97 ]

Free(ptr[0])

returned 0

Free List [ Size 2 ]: [ addr:1000 sz:3 ][ addr:1003 sz:97 ]

ptr[1] = Alloc(5) returned 1003 (searched 2 elements)

Free List [ Size 2 ]: [ addr:1000 sz:3 ][ addr:1008 sz:92 ]

Free(ptr[1])

returned 0

Free List [ Size 3 ]: [ addr:1000 sz:3 ][ addr:1003 sz:5 ][ addr:1008 sz:92 ]

ptr[2] = Alloc(8) returned 1008 (searched 3 elements)

Free List [ Size 3 ]: [ addr:1000 sz:3 ][ addr:1003 sz:5 ][ addr:1016 sz:84 ]

Free(ptr[2])

returned 0

Free List [ Size 4 ]: [ addr:1000 sz:3 ][ addr:1003 sz:5 ][ addr:1008 sz:8 ][ addr:1016 sz:84 ]

ptr[3] = Alloc(8) returned 1008 (searched 3 elements)

Free List [ Size 3 ]: [ addr:1000 sz:3 ][ addr:1003 sz:5 ][ addr:1016 sz:84 ]

Free(ptr[3])

returned 0

Free List [ Size 4 ]: [ addr:1000 sz:3 ][ addr:1003 sz:5 ][ addr:1008 sz:8 ][ addr:1016 sz:84 ]

ptr[4] = Alloc(2) returned 1000 (searched 1 elements)

Free List [ Size 4 ]: [ addr:1002 sz:1 ][ addr:1003 sz:5 ][ addr:1008 sz:8 ][ addr:1016 sz:84 ]

ptr[5] = Alloc(7) returned 1008 (searched 3 elements)

Free List [ Size 4 ]: [ addr:1002 sz:1 ][ addr:1003 sz:5 ][ addr:1015 sz:1 ][ addr:1016 sz:84 ]

**malloc.py -n 10 -H 0 -p FIRST -s 0 -l SIZESORT-**

ptr[0] = Alloc(3) returned 1000 (searched 1 elements)

Free List [ Size 1 ]: [ addr:1003 sz:97 ]

Free(ptr[0])

returned 0

Free List [ Size 2 ]: [ addr:1003 sz:97 ][ addr:1000 sz:3 ]

ptr[1] = Alloc(5) returned 1003 (searched 1 elements)

Free List [ Size 2 ]: [ addr:1008 sz:92 ][ addr:1000 sz:3 ]

Free(ptr[1])

returned 0

Free List [ Size 3 ]: [ addr:1008 sz:92 ][ addr:1003 sz:5 ][ addr:1000 sz:3 ]

ptr[2] = Alloc(8) returned 1008 (searched 1 elements)

Free List [ Size 3 ]: [ addr:1016 sz:84 ][ addr:1003 sz:5 ][ addr:1000 sz:3 ]

Free(ptr[2])

returned 0

Free List [ Size 4 ]: [ addr:1016 sz:84 ][ addr:1008 sz:8 ][ addr:1003 sz:5 ][ addr:1000 sz:3 ]

ptr[3] = Alloc(8) returned 1016 (searched 1 elements)

Free List [ Size 4 ]: [ addr:1024 sz:76 ][ addr:1008 sz:8 ][ addr:1003 sz:5 ][ addr:1000 sz:3 ]

Free(ptr[3])

returned 0

Free List [ Size 5 ]: [ addr:1024 sz:76 ][ addr:1008 sz:8 ][ addr:1016 sz:8 ][ addr:1003 sz:5 ][ addr:1000 sz:3 ]

ptr[4] = Alloc(2) returned 1024 (searched 1 elements)

Free List [ Size 5 ]: [ addr:1026 sz:74 ][ addr:1008 sz:8 ][ addr:1016 sz:8 ][ addr:1003 sz:5 ][ addr:1000 sz:3 ]

ptr[5] = Alloc(7) returned 1026 (searched 1 elements)

Free List [ Size 5 ]: [ addr:1033 sz:67 ][ addr:1008 sz:8 ][ addr:1016 sz:8 ][ addr:1003 sz:5 ][ addr:1000 sz:3 ]

1. **随着时间推移，如果没有合并机制，内存容易被分割成小的碎片，大型请求会无法满足，地址返回-1。没有合并的情况下在590次请求就无法满足，有合并的情况下仍然能能够继续运行。**

**最终不合并的空闲列表大小为82，分成31段，合并的列表为1段98。**

**列表的排序仍然重要，排序可以减少搜索大量小碎片 （如大小为1），加快搜索速度。**

**测试代码：malloc.py -n 1000 -C -c**

Free(ptr[588])

returned 0

Free List [ Size 31 ]: [ addr:1000 sz:2 ][ addr:1002 sz:1 ][ addr:1006 sz:1 ][ addr:1007 sz:1 ][ addr:1008 sz:5 ][ addr:1013 sz:1 ][ addr:1014 sz:1 ][ addr:1015 sz:1 ][ addr:1016 sz:5 ][ addr:1021 sz:1 ][ a

ddr:1022 sz:3 ][ addr:1031 sz:1 ][ addr:1032 sz:2 ][ addr:1034 sz:3 ][ addr:1037 sz:4 ][ addr:1041 sz:1 ][ addr:1042 sz:2 ][ addr:1052 sz:1 ][ addr:1053 sz:6 ][ addr:1059 sz:2 ][ addr:1061 sz:1 ][ addr:1068

sz:1 ][ addr:1069 sz:3 ][ addr:1072 sz:5 ][ addr:1077 sz:3 ][ addr:1080 sz:1 ][ addr:1081 sz:5 ][ addr:1086 sz:3 ][ addr:1089 sz:5 ][ addr:1094 sz:2 ][ addr:1096 sz:4 ]

ptr[590] = Alloc(9) returned -1 (searched 31 elements)

Free List [ Size 31 ]: [ addr:1000 sz:2 ][ addr:1002 sz:1 ][ addr:1006 sz:1 ][ addr:1007 sz:1 ][ addr:1008 sz:5 ][ addr:1013 sz:1 ][ addr:1014 sz:1 ][ addr:1015 sz:1 ][ addr:1016 sz:5 ][ addr:1021 sz:1 ][ a

ddr:1022 sz:3 ][ addr:1031 sz:1 ][ addr:1032 sz:2 ][ addr:1034 sz:3 ][ addr:1037 sz:4 ][ addr:1041 sz:1 ][ addr:1042 sz:2 ][ addr:1052 sz:1 ][ addr:1053 sz:6 ][ addr:1059 sz:2 ][ addr:1061 sz:1 ][ addr:1068

sz:1 ][ addr:1069 sz:3 ][ addr:1072 sz:5 ][ addr:1077 sz:3 ][ addr:1080 sz:1 ][ addr:1081 sz:5 ][ addr:1086 sz:3 ][ addr:1089 sz:5 ][ addr:1094 sz:2 ][ addr:1096 sz:4 ]

**malloc.py -n 1000 -C -c**

malloc.py -n 1000 -C -c

Free(ptr[511])

returned 0

Free List [ Size 1 ]: [ addr:1000 sz:100 ]

ptr[514] = Alloc(2) returned 1000 (searched 1 elements)

Free List [ Size 1 ]: [ addr:1002 sz:98 ]

1. **高于50时，比起释放会更多的进行调用内存，越靠近100获取越多。在100时仅进行获取，不再进行释放。在接近0时每次获取后都会立即进行释放。（0会报错）**

**测试代码：malloc.py -n 10 -H 0 -p BEST -s 0 -P 60 -c**

**malloc.py -n 10 -H 0 -p BEST -s 0 -P 100 -c**

**malloc.py -n 10 -H 0 -p BEST -s 0 -P 0 -c**

(base) C:\Users\Michael Shi\Desktop\ostep-homework-dev-zhongyl\vm-freespace>malloc.py -n 10 -H 0 -p BEST -s 0 -P 60 -c

seed 0

size 100

baseAddr 1000

headerSize 0

alignment -1

policy BEST

listOrder ADDRSORT

coalesce False

numOps 10

range 10

percentAlloc 60

allocList

compute True

ptr[0] = Alloc(3) returned 1000 (searched 1 elements)

Free List [ Size 1 ]: [ addr:1003 sz:97 ]

ptr[1] = Alloc(5) returned 1003 (searched 1 elements)

Free List [ Size 1 ]: [ addr:1008 sz:92 ]

Free(ptr[0])

returned 0

Free List [ Size 2 ]: [ addr:1000 sz:3 ][ addr:1008 sz:92 ]

ptr[2] = Alloc(6) returned 1008 (searched 2 elements)

Free List [ Size 2 ]: [ addr:1000 sz:3 ][ addr:1014 sz:86 ]

Free(ptr[2])

returned 0

Free List [ Size 3 ]: [ addr:1000 sz:3 ][ addr:1008 sz:6 ][ addr:1014 sz:86 ]

ptr[3] = Alloc(8) returned 1014 (searched 3 elements)

Free List [ Size 3 ]: [ addr:1000 sz:3 ][ addr:1008 sz:6 ][ addr:1022 sz:78 ]

Free(ptr[1])

returned 0

Free List [ Size 4 ]: [ addr:1000 sz:3 ][ addr:1003 sz:5 ][ addr:1008 sz:6 ][ addr:1022 sz:78 ]

Free(ptr[3])

returned 0

Free List [ Size 5 ]: [ addr:1000 sz:3 ][ addr:1003 sz:5 ][ addr:1008 sz:6 ][ addr:1014 sz:8 ][ addr:1022 sz:78 ]

ptr[4] = Alloc(8) returned 1014 (searched 5 elements)

Free List [ Size 4 ]: [ addr:1000 sz:3 ][ addr:1003 sz:5 ][ addr:1008 sz:6 ][ addr:1022 sz:78 ]

Free(ptr[4])

returned 0

Free List [ Size 5 ]: [ addr:1000 sz:3 ][ addr:1003 sz:5 ][ addr:1008 sz:6 ][ addr:1014 sz:8 ][ addr:1022 sz:78 ]

**malloc.py -n 10 -H 0 -p BEST -s 0 -P 100 -c**

(base) C:\Users\Michael Shi\Desktop\ostep-homework-dev-zhongyl\vm-freespace>malloc.py -n 10 -H 0 -p BEST -s 0 -P 100 -c

seed 0

size 100

baseAddr 1000

headerSize 0

alignment -1

policy BEST

listOrder ADDRSORT

coalesce False

numOps 10

range 10

percentAlloc 100

allocList

compute True

ptr[0] = Alloc(8) returned 1000 (searched 1 elements)

Free List [ Size 1 ]: [ addr:1008 sz:92 ]

ptr[1] = Alloc(3) returned 1008 (searched 1 elements)

Free List [ Size 1 ]: [ addr:1011 sz:89 ]

ptr[2] = Alloc(5) returned 1011 (searched 1 elements)

Free List [ Size 1 ]: [ addr:1016 sz:84 ]

ptr[3] = Alloc(4) returned 1016 (searched 1 elements)

Free List [ Size 1 ]: [ addr:1020 sz:80 ]

ptr[4] = Alloc(6) returned 1020 (searched 1 elements)

Free List [ Size 1 ]: [ addr:1026 sz:74 ]

ptr[5] = Alloc(6) returned 1026 (searched 1 elements)

Free List [ Size 1 ]: [ addr:1032 sz:68 ]

ptr[6] = Alloc(8) returned 1032 (searched 1 elements)

Free List [ Size 1 ]: [ addr:1040 sz:60 ]

ptr[7] = Alloc(3) returned 1040 (searched 1 elements)

Free List [ Size 1 ]: [ addr:1043 sz:57 ]

ptr[8] = Alloc(10) returned 1043 (searched 1 elements)

Free List [ Size 1 ]: [ addr:1053 sz:47 ]

ptr[9] = Alloc(10) returned 1053 (searched 1 elements)

Free List [ Size 1 ]: [ addr:1063 sz:37 ]

**malloc.py -n 10 -H 0 -p BEST -s 0 -P 1 -c**

(base) C:\Users\Michael Shi\Desktop\ostep-homework-dev-zhongyl\vm-freespace>malloc.py -n 10 -H 0 -p BEST -s 0 -P 1 -c

seed 0

size 100

baseAddr 1000

headerSize 0

alignment -1

policy BEST

listOrder ADDRSORT

coalesce False

numOps 10

range 10

percentAlloc 1

allocList

compute True

ptr[0] = Alloc(5) returned 1000 (searched 1 elements)

Free List [ Size 1 ]: [ addr:1005 sz:95 ]

Free(ptr[0])

returned 0

Free List [ Size 2 ]: [ addr:1000 sz:5 ][ addr:1005 sz:95 ]

ptr[1] = Alloc(2) returned 1000 (searched 2 elements)

Free List [ Size 2 ]: [ addr:1002 sz:3 ][ addr:1005 sz:95 ]

Free(ptr[1])

returned 0

Free List [ Size 3 ]: [ addr:1000 sz:2 ][ addr:1002 sz:3 ][ addr:1005 sz:95 ]

ptr[2] = Alloc(9) returned 1005 (searched 3 elements)

Free List [ Size 3 ]: [ addr:1000 sz:2 ][ addr:1002 sz:3 ][ addr:1014 sz:86 ]

Free(ptr[2])

returned 0

Free List [ Size 4 ]: [ addr:1000 sz:2 ][ addr:1002 sz:3 ][ addr:1005 sz:9 ][ addr:1014 sz:86 ]

ptr[3] = Alloc(2) returned 1000 (searched 4 elements)

Free List [ Size 3 ]: [ addr:1002 sz:3 ][ addr:1005 sz:9 ][ addr:1014 sz:86 ]

Free(ptr[3])

returned 0

Free List [ Size 4 ]: [ addr:1000 sz:2 ][ addr:1002 sz:3 ][ addr:1005 sz:9 ][ addr:1014 sz:86 ]

ptr[4] = Alloc(4) returned 1005 (searched 4 elements)

Free List [ Size 4 ]: [ addr:1000 sz:2 ][ addr:1002 sz:3 ][ addr:1009 sz:5 ][ addr:1014 sz:86 ]

Free(ptr[4])

returned 0

Free List [ Size 5 ]: [ addr:1000 sz:2 ][ addr:1002 sz:3 ][ addr:1005 sz:4 ][ addr:1009 sz:5 ][ addr:1014 sz:86 ]

1. **使用最差适应算法申请大量空间大小为 1 的块然后释放，并且选择不合并即可。**

**测试代码：malloc.py -n 10 -H 0 -p BEST -s 0 -c -A +1,+1,+1,+1,+1,+1,+1,+1,+1,+1,+1,+1,+1,+1,+1,+1,+1,+1,+1,+1,+1,+1,+1,+1,+1,+1,+1,+1,+1,+1,+1,+1,+1,+1,+1,+1,+1,+1,+1,+1,+1,+1,+1,+1,+1,+1,+1,+1,+1,+1,+1,+1,+1,+1,+1,+1,+1,+1,+1,+1,+1,+1,+1,+1,+1,+1,+1,+1,+1,+1,+1,+1,+1,+1,+1,+1,+1,+1,+1,+1,+1,+1,+1,+1,+1,+1,+1,+1,+1,+1,+1,+1,+1,+1,+1,+1,+1,+1,+1,+1**

**ptr[97] = Alloc(1) returned 1097 (searched 1 elements)**

**Free List [ Size 1 ]: [ addr:1098 sz:2 ]**

**ptr[98] = Alloc(1) returned 1098 (searched 1 elements)**

**Free List [ Size 1 ]: [ addr:1099 sz:1 ]**

**ptr[99] = Alloc(1) returned 1099 (searched 1 elements)**

**Free List [ Size 0 ]:**