Implement my own shell report - 과제4

201521033 조경선

- Work description

본 과제는 Process Memory allocation을 구현하는 과제이다. 기본적으로 memory를 할당하기위해서는 heap 공간을 원하는 만큼 늘려주고 줄여준다. 여기서 heap은 mapped region, unmapped region 그리고 unusable region의 공간이 있는데, mapped region안에 메모리 block들을 할당시켜주고, unmapped region과 mapped region의 경계를 말해주는 brk를 움직여주면서 mapped region 공간을 원하는 만큼 변경시켜주는 것이 이 과제의 주 목적이다.

첫 번째로 alloc.h에 있는 meta structure에 prev, next, free, size 변수들을 지정해준다. 그리고 alloc.c에 있는 중요한 함수들을 구현하기 전에 메인 함수에서 file name을 입력시켜주면, 그 file을 읽어 들이고 file안에 있는 command들을 해석해주는 코드를 짰다.

```
20  int main()
21  {
22      FILE* file = NULL;
23      char filename[256];
24      char command_infile[256][256];
25
26      printf("Input the file name: ");
27      scanf("%s", filename);
28
29      printf("The file name is %s\n", filename);
30      printf("-------INPUT--------\n");
31
32      file = fopen(filename, "r");
33      if(file == NULL){
34          printf("ERROR: open file is empty.\n");
35          return 0;
36      }
37      else{
38          char *pStr;
39
40          while(1){
41              pStr = fgets(command_infile[n_line], 200, file);
42              if(feof(file)) break;
43              if(pStr == NULL){
44                  printf("ERROR: reading file.\n");
45                  return 0;
46              }
47              printf("%s\n", pStr);
48              n_line++;
49          }
50          fclose(file);
51      }
52      //read the file line by line
53
```

그림 1 main 함수

그림1과 같이 fgets 함수를 이용하여 file 안에 내용들을 line by line으로 command_infile[] list에 저장을 해준다. 그리고 첫번째 코멘드는 코멘드들의 개수와 fit type을 의미하므로, 이를 통해 meta structure를 초기화 시켜준다. 본인은 전역변수로 meta i_meta[10]을 선언하여 구조체 리스트를 생성했다.

```
84  void init_meta(char* command, size_t n_line){
85      //printf("line [0]: %s\n", command);
86
87      n_command = atoi(strtok(command , " \n\t"));
88      //printf("the # of commands: %d\n",n_command);
89      type_fit = strtok(NULL, " \n\t");
90      //printf("fit type: %s\n", type_fit);
91
92      if(n_command == 0){
93          printf("ERROR: nothing in file.\n");
94          return;
95      }
96      else{
97          for(int i=0; i<n_command; i++){
98              if(i == 0){
99                  i_meta[0].prev = NULL;
100                 i_meta[0].next = i_meta[1].prev;
101                 i_meta[0].free = 1; //false
102                 i_meta[0].size = 0;
103             }
104             else{
105                 i_meta[i].prev = i_meta[i-1].next;
106                 i_meta[i].next = NULL;
107                 i_meta[i].free = 1; //false
108                 i_meta[i].size = 0;
109             }
110         }
111
112     }
113     return;
114 }
```

그림 2 init_meta 함수

Meta 구조체를 코멘드의 수만큼 만들고, 코멘드 해석과 메모리를 할당을 시켜 주기위해 do_command 함수를 사용하여 alloc.c에 있는 구현하고자 하는 중심의 함수들을 부른다. Do_command 함수는 일단 들어온 command의 종류를 살펴보고 이를 분류해 각각 알맞은 함수로 넣어주었다. 그리고 그림 4는 alloc.c를 구현한 것인데, meta structure를 쓰기 위해 extern을 사용하여 alloc.c에서도 사용 가능하도록 하였다. M_malloc함수에서는 meta 구조체의 내용을 바꾸고, meta size를 size/4*4+4로 설정하였다. 이는 할당해주는 메모리 사이즈의 단위를 4바이트 단위로 끊어 주기 위함이다. 그 후 sbrk 함수를 이용하여 요구한 크기만큼 heap의 크기를 늘려주고 그 주소값을 리턴해준다. M_free 함수 같은 경우는 완벽히 구현하지는 못했다. free시켜주고자하는 메모리의 주소값을 받아 그 주소값에 있는 스트링을 없애주고, 과제의 요구사항에 따라 memory를 다시 돌려주는 것이 아니라 전후 블락들의 상태를 보고 메모리를 합칠 것인지 없앨 것인지 구현하는 것이 최종 목표였다. 하지만 본인의 역량 부족으로 그림4와 같이 짜는 것으로 과제를 마무리 지었다.

```
116  void do_command(char* command, size_t k){
117      //printf("line: %s\n", command);
118
119      char* command_type;
120      char* content;
121      unsigned int content_size;
122      int n,m;
123
124      command_type = strtok(command , " \n\t");
125      //printf("the type of commands: %s\n",command_type);
126
127
128      switch(*command_type){
129          case 's':
130          if(i_meta[k].free == 1){
131              content = strtok(NULL, "\n");
132              addr[k] = (char*)m_malloc(strlen(content));
133              strcpy(addr[k], content);
134              t++;
135          }
136          break;
137          case 'f':
138          n = atoi(strtok(NULL, "\n"));
139          m_free(addr[n]);
140          if(strcmp(addr[n],"")==0){
141              i_meta[n].free = 1;
142              i_meta[n].next = NULL;
143              //change info about the one which is free
144              i_meta[k].prev = NULL;
145              i_meta[k].free = NULL;
146              i_meta[k].size = NULL;
147              //info about f command line
148          }
149          t++;
150          break;
151          case 'r':
152          n = atoi(strtok(NULL, " \n\t"));
153          m = atoi(strtok(NULL, "\n"));
154          printf("the type of commands: %c\n",*command_type);
155          break;
156          case 'e':
157          n = atoi(strtok(NULL, "\n"));
158          printf("the type of commands: %c\n",*command_type);
159          break;
160          default:
161          printf("There is no command like that.\n");
162          break;
163
164      }
165
166      return;
167  }
```

그림 3 do_command 함수

```c
extern meta i_meta[10];
extern int t;

void *m_malloc(size_t size){

    if(size == 0)
        return NULL;
    else{
        //printf("malloc size: %d\n", size);
        //printf("t: %d\n", t);
        i_meta[t].free = 0;//true
        i_meta[t].size = size/4*4+4;//aligning the requested size in 4 bytes
        //printf("size: %d\n", i_meta[t].size);
        unsigned int pBrk = sbrk(i_meta[t].size);
        if(pBrk == 0){
            printf("ERROR: sbrk\n");
            return NULL;
        }
        else{
            //printf("brk: %d\n", pBrk);
            return pBrk;
        }
    }

}//Allocation

void m_free(void *ptr){
    if(strcmp(ptr,"")==0){
        return;
    }
    strcpy(ptr,"");
    if(i_meta[t].prev != NULL && i_meta[t-1].free == 1){
        //fusion the two blocks
    }
    if(i_meta[t].next != NULL && i_meta[t+1].free == 1){
        //fusion the two blocks
    }
    if(i_meta[t].next == NULL){
        //release memory
    }
    return;

}//Deallocation

```

그림 4 alloc.c

본인은 directory input 과 inpu2를 만들어서 pdf와 example 파일에 나와있던 처음 두 예시를 실행시켜보며 테스트해보았다. 두 예시를 테스트해 본 결과는 그림 5와 6에 나와있다.

```
ajou@ajou-VirtualBox:~/1801_OS_assignment4$ ./main
Input the file name: input
The file name is input
-------INPUT--------
3 F

s Think like a man of action and act like man of thought.

s Courage is very important. Like a muscle, it is strengthened by use.

s Life is the art of drawing sufficient conclusions from insufficient premises.

--------OUTPUT--------
0 56 Think like a man of action and act like man of thought.
0 72 Courage is very important. Like a muscle, it is strengthened by use.
0 80 Life is the art of drawing sufficient conclusions from insufficient premise
s.
```

그림 5 example 1

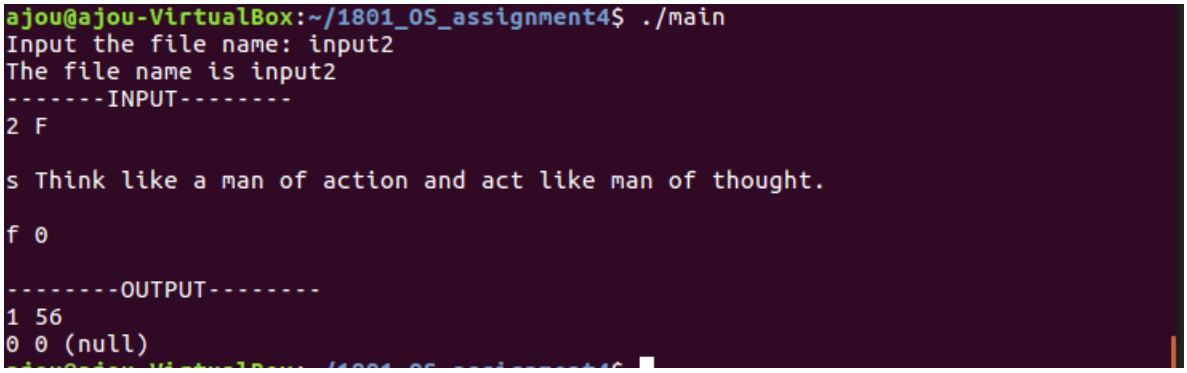

그림 6 example 2

- Lessons

이번 과제에서 heap을 manage하는 법, 주소를 할당시켜 주는 것, sbrk의 사용법에 대해 공부했다. 그리고 heap이 메모리를 어떻게 할당해주는지 정확히 알게 되었다.

- Feedback

이번에는 과제 1,2,3보다 초반에는 수월하게 코딩을 했다. main함수의 구조는 비교적 간단해서 차근차근 구현을 할 수 있었지만, 중요한 함수들을 구현하는 것은 쉽지 않았다. Meta 구조체를 alloc.c와 main.c에서도 써야 했는데, 초기화를 main 함수에서 진행하다 보니 alloc.c에서 구조체의 내용을 바꿔주기 쉽지 않았다. 본인은 extern을 사용하여 meta 구조체를 변형했지만, 다른 더 좋은 방법이 있을 것이라고 생각한다. 또한, 구조체 리스트와 할당된 메모리가 같은 heap에 있다는 것을 확신하지 못하였다. 또한, 만약 그러한 구조가 아니라면 reallocation에서 메모리를 합치는 방법이 무엇인지 고민해보아도 쉽게 생각이 나질 않았다. 많은 아쉬움이 남는 과제지만, 다음에는 꼭 성공을 시켜보고 싶은 과제이다.