1. 程式碼

(1) 標頭檔及全域變數、函式

```c
#include <stdio.h>
#include <stdlib.h>
#include <pwd.h>
#include <sys/types.h>
#include <sys/wait.h>
#include <unistd.h>
#include <string.h>
#include <termios.h>
#include <stdbool.h>
#include <signal.h>
#define MAXSIZE 64
pid_t pid;
char **history;
int num_command=0;
int bg_id[MAXSIZE];
int num_bg=0;
int buildin_command(char**);
int getch();
void prompt();
void process();
void background();
void catch_CHLD(int sig_num){
        signal(SIGCHLD,catch_CHLD);
        for(int j=0;j<num_command;j++){
                if(bg_id[j]==pid)
                        bg_id[j]=0;
        }
```

(2) Main function

```c
int main(){
        history=(char**)malloc(MAXSIZE*sizeof(char*));
        while(1){
                prompt();
                signal(SIGCHLD,catch_CHLD);
                process();
        }
}
```

(3) Prompt(): 印出 prompt

```
void prompt(){
        char hostname[MAXSIZE];
        char current_dir[MAXSIZE];
        char *home_path;
        int home_path_len;
        struct passwd *user;
        user=getpwuid(getuid());//get data from passwd using given uid
        home_path=user->pw_dir;
        home_path_len=strlen(home_path);
        gethostname(hostname,sizeof(hostname));
        getcwd(current_dir,sizeof(current_dir));
        if(!strncmp(home_path,current_dir,home_path_len)){//shorten path name u
sing ~
                current_dir[0]='~';
                for(int i=home_path_len;i<MAXSIZE;i++)
                        current_dir[i-home_path_len+1]=current_dir[i];
                for(int i=home_path_len+1;i<MAXSIZE;i++)
                        current_dir[i]='\0';
        }
        printf("%s@%s:%s",user->pw_name,hostname,current_dir);
        if(geteuid()==0)
                printf("# ");//represent super user
        else
                printf("$ ");//represent normal user
}
```

(4) Signal(SIGCHLD,catch_CHLD): 使用 signal 處理 SIGCHLD(用於 background execution)

```
void catch_CHLD(int sig_num){
        signal(SIGCHLD,catch_CHLD);
        for(int j=0;j<num_command;j++){
                if(bg_id[j]==pid)
                        bg_id[j]=0;
        }
}
```

(5) Process(): 讀入指令，判斷其為內部指令、外部指令或查詢歷史 command，並判斷是否需要進行 background execution

```
void process(){
        char *str;
        char **instruction;
        int index=num_command;
        int c;
        int tmp=0;
        bool background=false;//whether it needs to execute in background
        str=(char*)malloc(MAXSIZE*sizeof(char));
        while(1){
                c=getch();
                if(c==27){//represent ^ (up and down)(be used in finding histor
y command
                        c=getch();//represent [
                        c=getch();
                        if(c==65){//up (^[A)
                                if(index-1<0)
                                        index=0;
                                else
                                        index--;
                        }
                        else if(c==66){//down (^[B)
                                if(index+1>=num_command)
                                        index=num_command-1;
                                else
                                        index++;
                        }
                        str=history[index];
```

```
                        printf("\033[%dD",MAXSIZE);//modify the prompt line
                        for(int j=0;j<=MAXSIZE;j++)
                                printf(" ");
                        printf("\033[%dD",MAXSIZE);
                        printf("\r");
                        prompt();
                        if(str!=NULL){
                                printf("%s",str);
                                for(int j=0;j<MAXSIZE;j++){
                                        if(str[j]=='&'){
                                                str[j]='\0';
                                                background=true;
                                                break;
                                        }
                                }
                        }
                }
                else if(c==10){//represent enter
                        printf("\n");
                        history[num_command]=(char*)malloc(MAXSIZE*sizeof(char)
);
                        strcpy(history[num_command],str);
                        if(background)
                                strcat(history[num_command],"&");
                        num_command++;
```

```
                        break;
                }
                else if(c==127){//represent backspace
                        str[--tmp]='\0';
                        printf("\033[D \033[D");
                        continue;
                }
                else{
                        if(((char)c)=='&')
                                background=true;
                        else
                                str[tmp++]=(char)c;
                        printf("%c",c);
                }
        }
        int i=0;
        instruction=(char**)malloc(MAXSIZE*sizeof(char*));
        instruction[i]=strtok(str," ");//seperate the string using blank
        while(instruction[i]!=NULL){
                instruction[++i]=strtok(NULL," ");
        }
        if(instruction[0]==NULL)
                return;
        else if(buildin_command(instruction))//judge if it's a build in command
                return;
        pid=fork();
        if(pid==0){
```

```
        }
        else{
                if(background==false)
                        wait(NULL);
                else{
                        bg_id[num_bg++]=pid;
                        printf("[%d] %d\n",num_bg,pid);
                }
        }
        return;
}
```

(6) Buildin_command(): 如果是內部指令則在此處理、輸出並回傳 1，如果是外部指令則回傳 0 (bg 在這裡處理)

```
int buildin_command(char **instruction){
        if(!strcmp(instruction[0],"export")){//export
                char *var=strtok(instruction[1],"=");
                char *env=strtok(NULL,":");
                env=strtok(NULL,"\"");
                setenv(var,env,1);
                return 1;
        }
        else if(!strcmp(instruction[0],"echo")){//echo
                for(int i=1;i<MAXSIZE;i++){
                        if(instruction[i]==NULL)
                                break;
                        if(instruction[i][0]=='$'){
                                char *buf;
                                buf=(char*)malloc(MAXSIZE*(sizeof(char)));
                                char *env;
                                for(int j=1;j<MAXSIZE;j++)
                                        buf[j-1]=instruction[i][j];
                                buf[MAXSIZE-1]='\0';
                                env=getenv(buf);
                                if(env==NULL)
                                        continue;
                                else
                                        printf("%s ",env);
                        }
                        else
                                printf("%s ",instruction[i]);
                printf("\n");
                return 1;
        }
        else if(!strcmp(instruction[0],"pwd")){//pwd
                char buf[MAXSIZE];
                getcwd(buf,sizeof(buf));
                printf("%s\n",buf);
                return 1;
        }
        else if(!strcmp(instruction[0],"cd")){//cd
                if(chdir(instruction[1])==-1)
                        printf("cd: %s: No such file or directory\n",instructio
n[1]);
                return 1;
        }
        else if(!strcmp(instruction[0],"bg")){
                int unfinish=0;
                for(int j=0;j<num_bg;j++){
                        if(bg_id[j]>0)
                                unfinish++;
                        else if(bg_id[j]==0){
                                printf("[%d] Done \n",j+1);
                                bg_id[j]=-1;
                        }
                }
                if(unfinish==0){
                        num_bg=0;
```

```
                                bg_id[j]=-1;
                        }
                }
                if(unfinish==0){
                        num_bg=0;
                        printf("bg: job has terminated\n");
                }
                else
                        printf("bg: job %d already in background\n",unfinish);
        }
        else
                return 0;
}
```

(7) Getch(): 從鍵盤讀取單個字符，不須等待回車鍵即可返回輸入字符的 ascii code

```
int getch()
{
    struct termios oldattr, newattr;
    int ch;
    tcgetattr( STDIN_FILENO, &oldattr );
    newattr = oldattr;
    newattr.c_lflag &= ~( ICANON | ECHO );
    tcsetattr( STDIN_FILENO, TCSANOW, &newattr );
    ch = getchar();
    tcsetattr( STDIN_FILENO, TCSANOW, &oldattr );
    return ch;
}
```

2. 基本功能(內部指令)

(1) Echo

```
cs4108056051@cs4108056051-VirtualBox:~$ ./hw1
cs4108056051@cs4108056051-VirtualBox:~$ echo 123
123
cs4108056051@cs4108056051-VirtualBox:~$ echo $PATH
/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin:/usr/games:/usr/lo
cal/games:/snap/bin
```

(2) Export

```
cs4108056051@cs4108056051-VirtualBox:~$ export PATH="$PATH:/home/cs4108056051"
cs4108056051@cs4108056051-VirtualBox:~$ echo $PATH
/home/cs4108056051
```

(3) Pwd

```
cs4108056051@cs4108056051-VirtualBox:~$ pwd
/home/cs4108056051
```

(4) Cd

```
cs4108056051@cs4108056051-VirtualBox:~$ cd lab12
cs4108056051@cs4108056051-VirtualBox:~/lab12$ pwd
/home/cs4108056051/lab12
```

3. 基本功能(外部指令)

(1) Ls

```
cs4108056051@cs4108056051-VirtualBox:~$ ls
client            hw1.c         lab6-2.c            Public      test.c
client.c          kernelModule  lab9                receiver    test_getch
consumer          lab10         linux               receiver.c  test_getch.c
consumer.c        lab10-1       linux-5.4.2         sender      testModule
Desktop           lab11         linux-5.4.2.tar.xz  sender.c    test.sh
Documents         lab12         Music               server      testSocket
Downloads         lab6-1        Pictures            server.c    Videos
examples.desktop  lab6-1.c      producer            Templates
hw1               lab6-2        producer.c          test
```

(2) Vim and cp

```
cs4108056051@cs4108056051-VirtualBox:~$ cp hw1.c hw2.c
cs4108056051@cs4108056051-VirtualBox:~$ ls
client            hw1.c         lab6-2              producer.c  test
client.c          hw2.c         lab6-2.c            Public      test.c
consumer          kernelModule  lab9                receiver    test_getch
consumer.c        lab10         linux               receiver.c  test_getch.c
Desktop           lab10-1       linux-5.4.2         sender      testModule
Documents         lab11         linux-5.4.2.tar.xz  sender.c    test.sh
Downloads         lab12         Music               server      testSocket
examples.desktop  lab6-1        Pictures            server.c    Videos
hw1               lab6-1.c      producer            Templates
cs4108056051@cs4108056051-VirtualBox:~$ vim hw1.c
cs4108056051@cs4108056051-VirtualBox:~$ vim hw2.c
```

```
#include <stdio.h>
#include <stdlib.h>
#include <pwd.h>
#include <sys/types.h>
#include <sys/wait.h>
#include <unistd.h>
#include <string.h>
#include <termios.h>
#include <stdbool.h>
#include <signal.h>
#define MAXSIZE 64
pid_t pid;
char **history;
int num_command=0;
int bg_id[MAXSIZE];
int num_bg=0;
int buildin_command(char**);
int getch();
void prompt();
void process();
void background();
void catch_CHLD(int sig_num){
        signal(SIGCHLD,catch_CHLD);
        for(int j=0;j<num_command;j++){
                if(bg_id[j]==pid)
                        bg_id[j]=0;
        }
"hw1.c" 223L, 4923C                                   4,19          Top
```

```c
#include <stdio.h>
#include <stdlib.h>
#include <pwd.h>
#include <sys/types.h>
#include <sys/wait.h>
#include <unistd.h>
#include <string.h>
#include <termios.h>
#include <stdbool.h>
#include <signal.h>
#define MAXSIZE 64
pid_t pid;
char **history;
int num_command=0;
int bg_id[MAXSIZE];
int num_bg=0;
int buildin_command(char**);
int getch();
void prompt();
void process();
void background();
void catch_CHLD(int sig_num){
        signal(SIGCHLD,catch_CHLD);
        for(int j=0;j<num_command;j++){
                if(bg_id[j]==pid)
                        bg_id[j]=0;
        }
"hw2.c" 223L, 4923C                                           1,1           Top
```

(3) Rm

```
cs4108056051@cs4108056051-VirtualBox:~$ rm hw2.c
cs4108056051@cs4108056051-VirtualBox:~$ ls
client            hw1.c         lab6-2.c            Public      test.c
client.c          kernelModule  lab9                receiver    test_getch
consumer          lab10         linux               receiver.c  test_getch.c
consumer.c        lab10-1       linux-5.4.2         sender      testModule
Desktop           lab11         linux-5.4.2.tar.xz  sender.c    test.sh
Documents         lab12         Music               server      testSocket
Downloads         lab6-1        Pictures            server.c    Videos
examples.desktop  lab6-1.c      producer            Templates
hw1               lab6-2        producer.c          test
```

4. 進階功能

(1) 查詢歷史 command

(影片)(壓縮檔內的查詢歷史 command.mp4)

```
cs4108056051@cs4108056051-VirtualBox:~$ ./hw1
cs4108056051@cs4108056051-VirtualBox:~$ pwd
/home/cs4108056051
cs4108056051@cs4108056051-VirtualBox:~$ cd lab12
cs4108056051@cs4108056051-VirtualBox:~/lab12$ cd ..
cs4108056051@cs4108056051-VirtualBox:~$ pwd
/home/cs4108056051
cs4108056051@cs4108056051-VirtualBox:~$ cd lab12
cs4108056051@cs4108056051-VirtualBox:~/lab12$ pwd
/home/cs4108056051/lab12
```

(2) Background execution

```
cs4108056051@cs4108056051-VirtualBox:~$ ./hw1
cs4108056051@cs4108056051-VirtualBox:~$ sleep 5 &
[1] 6930
cs4108056051@cs4108056051-VirtualBox:~$ sleep 5 &
[2] 6931
cs4108056051@cs4108056051-VirtualBox:~$ bg
[1] Done
bg: job 1 already in background
cs4108056051@cs4108056051-VirtualBox:~$ bg
[2] Done
bg: job has terminated
```

(3) Output redirection

```
 for(int j=0;j<MAXSIZE;j++){
         if(!strcmp(instruction[j],">")){
                 redirect(instruction);
                 return;
         }
 }
```

```
void redirect(char **instruction){
        char *output;
        char **command;
        command=(char**)malloc(MAXSIZE*sizeof(char*));
        for(int j=0;j<MAXSIZE;j++){
                if(!strcmp(instruction[j],">")){
                        output=instruction[j+1];
                        break;
                }
                else{
                        command[j]=instruction[j];
                }
        }
        int fd=open(output,O_RDWR | O_CREAT,0644);
        int fd1;
        dup2(1,fd1);
        dup2(fd,1);
        buildin_command(command);
        fflush(stdout);
        dup2(fd1,1);
        prompt();
        close(fd1);
        return;
}
-- INSERT --                                              255,2         Bot
```

```
cs4108056051@cs4108056051-VirtualBox:~$ ./hw1
cs4108056051@cs4108056051-VirtualBox:~$ echo 123 > file

cs4108056051@cs4108056051-VirtualBox:~$ cat file
123
```