به نام خدا

راهنمای پروژه اول – ویراست نخست – ۱۴۰۰/۱۲/۲۸

در این مدرک سعی می کنم در مورد دو قسمت اول پروژه توضیح کافی ارائه کنم. اول به چگونگی ارتباط با I/O پرداخته و دوم ایجاد جدول سمبل ها را تشریح خواهم کرد.

۱. File I/O: در زبان برنامه نویسی سی، مانند هر زبان دیگر، می توان با برنامه در حال اجرا بصورت داینامیک ارتباط برقرار کرد. این ممکن است در شروع اجرای برنامه و یا در حین اجرا صورت گیرد. در آغاز اجرای برنامه دو متغیر در اختیار شما گذاشته شده به نامهای argc و argv که اولی از نوع عدد صحیح است و تعداد ارگمانهای ورودی خط اجرا را به برنامه بازمی گرداند. دومی از نوع char ** می باشد، ماتریسی از رشته های خط اجراست. به مثال زیر دقت کنید:

```
C:\home\mehran> assemble test.as test.mc
```

قسمت زرد رنگ همانا پرامپت سیستم است که روی سیستم شما متفاوت خواهد بود. بعد از علامت بزرگتر، سه رشته در خط اجرا وجود دارد، بنابراین argc برابر ۳ بوده و argv یک ماتریس دوبعدی از کاراکترها به صورت داده شده است:

assemble
test.as
test.mc

اکنون به قطعه کد شکل ۱ دقت کنید.

```
void main(int argc,char **argv){
  FILE *assp,*machp,*fopen();
   if(argc < 3){
     printf("***** Please run this program as follows:\n");
     printf("***** %s assprog.as machprog.m\n",argv[0]);
     printf("***** where assprog.as is your assembly program\n");
     printf("***** and machprog.m will be your machine code.\n");
     exit(1);
   }
   if((assp=fopen(argv[1],"r")) == NULL){
     printf("%s cannot be opened\n",argv[1]);
     exit(1);
   }
   if((machp=fopen(argv[2],"w+")) == NULL){
     printf("%s cannot be opened\n",argv[2]);
     exit(1);
   }
   // here you can place your code for the assembler
    fclose(assp);
    fclose(machp);
}
```

شکل ۱: قطعه کد برای ارتباط با فایلهای ورودی

در توضیح قطعه کد فوق تنها باید بگویم که ایف اول مطمئن می شود که کاربر با اجرای برنامه، دو فایل برنامه اسمبلی و برنامه ماشین را وارده کرده است. ایفهای دوم و سوم آنها (فایلها) را برای عملیات خواندن و نوشتن باز می کند و در عاقبت با fclose اصطلاحا فایلها کلوز می شوند.

۲. ایجاد جدول سمبل ها: برای این تسک اول یک ساختمان داده به صورت زیر تعریف کردم:

```
struct symbolTable{
   int value;
   char *symbol;
};
```

و با قطعه کد زیر که از دو تابع تشکیل شده است این جدول را ساختم.

```
symTabLen=findSymTabLen(assp);
pSymTab=(struct symbolTable *)malloc(symTabLen*sizeof(struct symbolTable));
for(i=0;i<symTabLen;i++)
   pSymTab[i].symbol=(char *)malloc(10);
fillSymTab(pSymTab,assp);
```

شاکله این دو تابع در شکل ۲ آمده است.

```
int findSymTabLen(FILE *inputFile){
  int count=0;
  size_t lineSize;
  char *line;
// write code to find the number of symbols in the assembly program
  free(line);
  return count;
}
void fillSymTab(struct symbolTable *symT,FILE *inputFile){
  int lineNo=0;
  size_t lineSize;
  char *line;
  int i=0;
//write code to construct the symbol table
  free(line);
}
```

شکل ۲: ساختار توابع مورد نیاز برای ایجاد جدول سمبل ها

در هر کدام از این دو تابع، نیاز است تا inputFile، خط به خط خوانده شده و عملیات مورد نظر روی هر خط از این فایل صورت می گیرد.

موفق باشید.