운영체제(3주차실습)

컴퓨터공학과 20135151 이갑성

|  |
| --- |
| 코드(주석)  <main.c>  #include <stdio.h>    #define SIZE 25  extern int\* getData(int \*pAry, int arySize);  extern void selectSort(int \*pAry, int \*last);  extern void printData(int \*pAry, int \*last);  int main(void)  {  int ary[SIZE];  int \*pLast;  pLast = getData(ary, SIZE);  selectSort(ary, pLast);  printData(ary, pLast);  return 0;  }    <get\_data.c>  #include <stdio.h>    int\* getData(int \*pAry, int arySize)  {  int ioResult;  int readCnt = 0;  int \*pFill = pAry;    //배열의 주소에 연결되있는 포인터 \*pFill은 값을 입력받고 성공적으로 입력받으면 인자의 개수를 리턴하기에 1을 리턴한다.(입력받는것을 실패하면 0을 리턴)  //조건문 if는 scanf함수에 리턴받은 값이 1이면 pFill의 위치를 이동시켜주고 readCnt의 값을 증가시켜준다.  //반복문 while은 ioRead의 값이 1이고, readCnt의 값이 arySize(20)보다 작을때까지 반복문을 수행한다.  do{  printf("Please enter number or <EOF>: ");  ioResult = scanf("%d", pFill);  if(ioResult == 1)  {  pFill++;  readCnt++;  }  }while(ioResult == 1 && readCnt < arySize);  printf("\n\n%d number read.", readCnt);  //값이 저장되어있는 배열을 가르키는 포인터 pFill의 위치를 마지막 값이 들어있는  곳으로하기 위해서 --을 해서 앞당겨 준다.  return(--pFill);  }  <exchange.c>  #include <stdio.h>    void exchange(int \*p1, int \*p2)  {  int temp;  //\*p1은 배열의 첫부분의 값(배열 0번째)을 가르키고, \*p2는 배열의 값 중에서 가장 작은 값을 가르키고 있다.  //temp에 \*p1을 넣어주고, \*p1에 가장 작은값이 들어있는 \*p2를 넣어준다. 그리고 다시 \*p2에 temp값을 넣어줌으로써  //\*p1의 값고 \*p2의 값을 바꿔준다.  temp = \*p1;  \*p1 = \*p2;  \*p2 = temp;  return;  }  <print\_data.c>  #include <stdio.h>    void printData(int \*pAry, int \*pLast)  {  int numPrt;  int \*pPrint;    //pPrint는 값이 들어있는 배열 pAry의 주소를 가르키는걸로 초기화하고, numPrt는 0으로 초기화 해준다  //pPrint의 주소가 pLast(배열의 마지막 주소)와 작거나 같을때까지 반복문을 수행한다.  //numPrt는 두자리수로 출력하고 공백 4칸을 띄어서 \*pPrint를 출력해준다  // numPrt 와 pPrint를 증가시켜준다.  printf("\n\nYour data sorted are: \n");  for(pPrint = pAry, numPrt = 0; pPrint <= pLast; numPrt++, pPrint++){  printf("\n#%02d %4d", numPrt, \*pPrint);  }  printf("\n\nEnd of List\n");  return;  }  ~  <select\_sort.c>  #include <stdio.h>    extern int\* smallest(int \*pAry, int \*pLast);  extern void exchange(int \*current, int \*smallest);  void selectSort(int \*pAry, int \*pLast)  {  int \*pWalker;  int \*pSmallest;  //pWalker은 배열의 첫번째 주소를 가르키게 초기화 해주고, pWalker이 마지막 주소를 가르키고 있는 pLast전 까지 pWalker의 주소를 증가시키며 반복문을 돌린다.  //smallest()함수에서 배열 중에서 가장 작은 값의 주소를 리턴해주므로 pSmallest에 배열의 가장 작은 값의 주소가 들어있다.  //exchange()함수를 이용하여 pWalker(배열의 첫번째 주소)의 값과 가장 작은값을 갖고 있는 pSmallest의 값을 바꿔준다.  //pWalker의 주소값을 증가되고 두번째 주소값을 가르키고 위와 동일한 방법으로 움직인다.  for(pWalker = pAry; pWalker < pLast; pWalker++){  pSmallest = smallest(pWalker, pLast);  exchange(pWalker, pSmallest);  }  return;  }  <smallest.c>  #include <stdio.h>    int\* smallest(int \*pAry, int \*pLast)  {  int \*pLooker;  int \*pSmallest;  //pSmallest는 배열중 첫번째의 주소로 초기화, pLooker은 배열의 첫번째보다 한칸  다음에 주소로 초기화  //pLooker은 끝에 있는 pLast까지 작거나 같을때까지 주소를 증가시켜 이동한다.  //조건문은 pLooker값이 pSmallest보다 작으면 pSmalles의 주소에 pLooker주소를  넣어준다.  for(pSmallest = pAry, pLooker = pAry+1; pLooker <= pLast; pLooker++)  {  if(\*pLooker < \*pSmallest)  {  pSmallest = pLooker;  }  }  return pSmallest;  } |
| 실행결과 |