REPORT #3



|  |
| --- |
| **제 출 일** : 2017년 09월 27일 |
| **과 목 명** : 컴퓨터프로그래밍및실습 |
| **담당교수 : 김정국 교수님** |
| **학 과** : 컴퓨터전자시스템 |
| **학 번** : 201703091 |
| **성 명 : 전기범** |

**HANKUK UNIVERSITY OF FOREIGN STUDIES**



**1 . 분수의 사칙연산 프로그램**

* 1. **문제 정의**

구조체 2개를 사용하며, 멤버 변수로 분자 분모를 각각 가짐, 사칙연산을 하는 함수 4개를 만들어 기약 분수 형태로 값을 return 하는 프로그램

**1.2. 문제 해결 방법**

구조체 2개를 사용, 멤버변수의 값과 연산자의 값을 입력 받은 후 strcmp를 사용하여 입력 받은 연산자가 어떤 연산을 하는지 구분, 해당 함수로 동작하도록 하였습니다. 또한 기약 분수는 분자와 분모의 최대공약수를 구하는 함수를 사용하여 구현하였습니다.

분모가 1일 경우 분자만 출력, 분모가 1이 아닌 경우 분자/분모 의 형태로 출력하게 하였습니다.

**1.3. 소스 코드**

|  |
| --- |
| #include <iostream>  #include <cstring> // strcmp사용  using namespace std;  int up, down; // 분자 분모 계산값, 접근의 용이성을 고려하여 전역변수로 선언함  void add(struct num1 a, struct num2 b);  void sub(struct num1 a, struct num2 b);  void multi(struct num1 a, struct num2 b);  void div(struct num1 a, struct num2 b);  int GCD(int num1, int num2);  struct num1  {  int up1;  int down1;  };  struct num2  {  int up2;  int down2;  };  int main()  {  char opr[6]; // 연산자  struct num1 a;  struct num2 b;  cin >> a.up1 >> a.down1;  cin >> opr;  cin >> b.up2 >> b.down2;  if (!strcmp(opr, "add")) // 입력된 연산자 확인 후 함수로 보냄  add(a, b);  else if (!strcmp(opr, "sub"))  sub(a, b);  else if (!strcmp(opr, "multi"))  multi(a, b);  else if (!strcmp(opr, "div"))  div(a, b);  int gcd = GCD(up, down);  up /= gcd, down /= gcd;  if (down == 1)  cout << up << endl; // 분모가 1이면 분자만 출력  else  cout << up << "/" << down << endl; // 분자 / 분모 출력  }  void add(struct num1 a, struct num2 b) // 덧셈 함수  {  int upper, lower; // 임시 저장용  upper = (a.up1\*b.down2) + (b.up2\*a.down1);  lower = a.down1\*b.down2;  up = upper;  down = lower;  }  void sub(struct num1 a, struct num2 b) // 뺄셈함수  {  int upper, lower;  upper = (a.up1\*b.down2) - (b.up2\*a.down1);  lower = a.down1\*b.down2;  up = upper;  down = lower;  }  void multi(struct num1 a, struct num2 b) // 곱셈함수  {  int upper, lower;  upper = a.up1\*b.up2;  lower = a.down1\*b.down2;  up = upper;  down = lower;  }  void div(struct num1 a, struct num2 b) // 나눗셈 함수  {  int upper, lower;  upper = a.up1\*b.down2;  lower = a.down1\*b.up2;  up = upper;  down = lower;  }  int GCD(int num1, int num2) // 분자와 분모의 최대공약수 구하기  {  int i, temp;  if (num1 >= num2)  temp = num2;  else  temp = num1;  for (i = temp; i>1; i--)  {  if (num1%i == 0 && num2%i == 0)  break;  }  return i;  } |

**1.4 결과 화면**

|  |
| --- |
|  |

**2 . 특정 조건까지 계속 입력 받는 프로그램**

**2.1 문제 정의**

**특정 조건이 입력되기 전까지 계속 입력을 받는 프로그램으로 동적할당으로 배열을 만들고 -2147483648이 입력될 경우 입력한 값들을 모두 합하여 출력하는 프로그램**

**2.2. 문제 해결 방법**

**배열의 동적 할당을 위하여 포인터 하나 선언, for문을 무한 반복하며 i의 값이 커질수록 배열의 크기도 커지게 설정, 입력 받은 배열의 값을 보존하기 위하여 임시 저장할 배열을 동적할당으로 선언하며 count라는 변수를 선언하여 몇번의 입력을 받았는지 측정, -2147483648이 입력되면 입력 받은 횟수만큼 값을 모두 더하고 출력하도록 하였습니다.**

**2.3. 소스 코드**

|  |
| --- |
| #include <iostream>  using namespace std;  int main()  {  int i, count = 0, calc = 0; // count는 몇번의 입력을 받았는지 확인하는 횟수, calc는 입력받은 배열의 합 저장  int \*arr = 0, \*temp = 0; // 동적할당을 위한 포인터  int stop = -2147483648LL; // -2147483648 입력시 프로그램 종료, long long형으로 형변환  for (i = 1;; i++)  {  arr = new int[i]; // i의 반복만큼 크기 동적할당  if (i > 1) // 1보다 클 경우 temp내용 arr로 옮김  for (int j = 0; j < i - 1; j++)  arr[j] = temp[j];  delete[] temp; // 사용 후 반납  temp = new int[i]; // i의 반복만큼 크기 동적할당  if (i > 1) // arr의 내용 temp에 저장  for (int j = 0; j < i - 1; j++)  temp[j] = arr[j];  cin >> arr[i - 1]; // arr 입력받음  if (arr[i - 1] == stop) // arr의 값이 -2147483648일 경우 입력 받은 값 모두 합산, for문 탈출  {  for (int i = 0; i < count; i++)  calc += temp[i];  break;  }  else // 그렇지 않으면 count +1, temp에 arr값 저장  {  count++;  temp[i - 1] = arr[i - 1];  delete[] arr; // 사용 후 반납  }  }  cout << calc << endl; // 합산한 값 출력  } |

**2.4 결과 화면**

|  |
| --- |
|  |

**3 . 학생 수 만큼 구조체 입력 받는 프로그램**

**3.1 문제 정의**

**학생의 수를 입력 받은 후 입력 받은 학생 수 만큼 구조체를 동적 할당하여 학생 수 만큼 멤버 변수들을 입력 받아 학생의 성적을 출력하는 프로그램**

**3.2. 문제 해결 방법**

학생의 수를 입력 받은 후 포인터를 사용하여 구조체의 크기를 학생 수 만큼 동적 할당, credit이라는 배열을 2칸 만들어 수학과 영어 점수의 합을 각각 저장 한 후 switch 문을 이용하여 수학과 영어 성적을 구조체 안 멤버 변수에 저장, 그 후 출력하였습니다.

**3.3. 소스 코드**

|  |
| --- |
| #include <iostream>  using namespace std;  struct student // 학생 구조체  {  char\*name;  int math[2];  char mathGrade;  int eng[2];  char engGrade;  };  int main()  {  int num; // 학생 수  struct student \*arr = 0; // 구조체 포인터  cin >> num;    arr = new student[num]; // 학생 수 만큼 구조체 배열 크기 동적할당  for (int i=0; i < num; i++)  {  int credit[2] = { 0 }; // 성적 저장 배열  arr[i].name = new char[100]; // 학생 이름 최대 100칸으로 동적할당  cin >> arr[i].name;  for (int j = 0; j < 2; j++)  {  cin >> arr[i].math[j]; // 수학 성적 중간 기말 순으로 입력  credit[0] += arr[i].math[j]; // 0번 배열에 수학 성적 합 저장  }  for (int j = 0; j < 2; j++)  {  cin >> arr[i].eng[j]; // 영어 성적 중간 기말 순으로 입력  credit[1] += arr[i].eng[j]; // 1번 배열에 영어 성적 합 저장  }  for (int n = 0; n < 2; n++)  {  switch (n) // 수학 = 0, 영어 = 1 구분  {  case 0:  if (credit[0] >= 180)  {  arr[i].mathGrade = 'A';  break;  }  else if ((credit[0] <= 179) && (credit[0] >= 160))  {  arr[i].mathGrade = 'B';  break;  }  else if ((credit[0] <= 159) && (credit[0] >= 130))  {  arr[i].mathGrade = 'C';  break;  }  else if ((credit[0] <= 129) && (credit[0] >= 100))  {  arr[i].mathGrade = 'D';  break;  }  else  {  arr[i].mathGrade = 'F';  break;  }  case 1:  if (credit[1] >= 180)  {  arr[i].engGrade = 'A';  break;  }  else if ((credit[1] <= 179) && (credit[1] >= 160))  {  arr[i].engGrade = 'B';  break;  }  else if ((credit[1] <= 159) && (credit[1] >= 130))  {  arr[i].engGrade = 'C';  break;  }  else if ((credit[1] <= 129) && (credit[1] >= 100))  {  arr[i].engGrade = 'D';  break;  }  else  {  arr[i].engGrade = 'F';  break;  }  }    }  }  for (int i = 0; i < num; i++)  {  cout << arr[i].name << " " << arr[i].mathGrade << " " << arr[i].engGrade << endl; // 출력  delete[] arr[i].name; // 출력했으면 반납  }  delete[] arr; // 실행 끝나면 반납  } |

**3.4 결과 화면**

|  |
| --- |
|  |

**4 . 특정 조건까지 구조체를 입력 받는 프로그램**

**4.1 문제 정의**

3번과 같은 프로그램으로 null을 입력 받았을 경우 입력을 종료하고 학생의 이름 및 성적을 출력하는 프로그램

**4.2. 문제 해결 방법**

2번과 3번의 해결방법을 합하여 해결하였습니다.

**4.3. 소스 코드**

|  |
| --- |
| #include <iostream>  #include <cstring> // strcmp사용을 위한 헤더  using namespace std;  struct student // 학생 구조체  {  char\*name;  int math[2];  char mathGrade;  int eng[2];  char engGrade;  };  int main()  {  int count = 0; // 몇 번 돌았는지 카운트  struct student \*arr = 0; // 구조체 포인터  struct student \*temp = 0; // 임시 저장 공간  for (int i = 1;; i++)  {  arr = new student[i]; // i의 크기만큼 동적 할당  if (i > 1) // arr에 temp의 값 넣기  for (int j = 0; j < i - 1; j++)  {  arr[j].name = temp[j].name;  arr[j].math[0] = temp[j].math[0];  arr[j].math[1] = temp[j].math[1];  arr[j].eng[0] = temp[j].eng[0];  arr[j].eng[1] = temp[j].eng[1];  arr[j].mathGrade = temp[j].mathGrade;  arr[j].engGrade = temp[j].engGrade;  }  temp = new student[i]; // i의 크기만큼 동적 할당  if (i > 1) // temp에 arr값 넣기  for (int j = 0; j < i - 1; j++)  {  temp[j].name = arr[j].name;  temp[j].math[0] = arr[j].math[0];  temp[j].math[1] = arr[j].math[1];  temp[j].eng[0] = arr[j].eng[0];  temp[j].eng[1] = arr[j].eng[1];  temp[j].mathGrade = arr[j].mathGrade;  temp[j].engGrade = arr[j].engGrade;  }  int credit[2] = { 0 }; // 성적 저장 배열  arr[i - 1].name = new char[100];// 학생 이름 최대 100칸으로 동적할당  cin >> arr[i - 1].name;  if (!strcmp(arr[i - 1].name, "null")) // null이 입력되면 for문 종료  break;  for (int j = 0; j < 2; j++)  {  cin >> arr[i - 1].math[j];// 수학 성적 중간 기말 순으로 입력  credit[0] += arr[i - 1].math[j]; // 0번 배열에 수학 성적 합 저장  }  for (int j = 0; j < 2; j++)  {  cin >> arr[i - 1].eng[j]; // 영어 성적 중간 기말 순으로 입력  credit[1] += arr[i - 1].eng[j]; // 1번 배열에 영어 성적 합 저장  }  // temp에 arr값 넣기  temp[i - 1].name = arr[i - 1].name;  temp[i - 1].math[0] = arr[i - 1].math[0];  temp[i - 1].math[1] = arr[i - 1].math[1];  temp[i - 1].eng[0] = arr[i - 1].eng[0];  temp[i - 1].eng[1] = arr[i - 1].eng[1];  temp[i - 1].mathGrade = arr[i - 1].mathGrade;  temp[i - 1].engGrade = arr[i - 1].engGrade;  for (int n = 0; n < 2; n++)  {  switch (n) // 수학 = 0, 영어 = 1 구분  {  case 0:  if (credit[0] >= 180)  {  temp[i - 1].mathGrade = 'A';  break;  }  else if ((credit[0] <= 179) && (credit[0] >= 160))  {  temp[i - 1].mathGrade = 'B';  break;  }  else if ((credit[0] <= 159) && (credit[0] >= 130))  {  temp[i - 1].mathGrade = 'C';  break;  }  else if ((credit[0] <= 129) && (credit[0] >= 100))  {  temp[i - 1].mathGrade = 'D';  break;  }  else  {  temp[i - 1].mathGrade = 'F';  break;  }  case 1:  if (credit[1] >= 180)  {  temp[i - 1].engGrade = 'A';  break;  }  else if ((credit[1] <= 179) && (credit[1] >= 160))  {  temp[i - 1].engGrade = 'B';  break;  }  else if ((credit[1] <= 159) && (credit[1] >= 130))  {  temp[i - 1].engGrade = 'C';  break;  }  else if ((credit[1] <= 129) && (credit[1] >= 100))  {  temp[i - 1].engGrade = 'D';  break;  }  else  {  temp[i - 1].engGrade = 'F';  break;  }  }  }  count++; // 카운트 증가  }  for (int i = 0; i < count; i++) // 카운트 만큼만 for문 반복  {  cout << arr[i].name << " " << temp[i].mathGrade << " " << temp[i].engGrade << endl; // 출력  delete[] arr[i].name; // 출력했으면 반납  }  delete[] arr; // 실행 끝나면 반납  delete[] temp; // 실행 끝나면 반납  } |

**4.4 결과 화면**

|  |
| --- |
|  |