REPORT #3

|  |
| --- |
| **제 출 일** : 2017년 9월 27일 |
| **과 목 명** : 컴퓨터 프로그래밍 및 실습 |
| **담당교수 : 김정국 교수님** |
| **학 과** : 컴퓨터 전자 시스템 공학부 |
| **학 번** : 201702234 |
| **성 명 : 유동혁** |

**HANKUK UNIVERSITY OF FOREIGN STUDIES**

****

**1. 분수의 사칙연산 프로그램**

* 1. **문제 정의**

-> 분수끼리 사칙연산을 하고 기약분수로 출력하는 프로그램

**1.2 문제 해결 방법**

-> 분수의 구조체 에는 분자, 분모를 저장한다. 기약분수로 바꾸는 법은 분수의 분자와 분모의 최대 공약수로 각각을 나눈다.

더하는 함수: 두 분모를 둘의 최소 공배수로 바꾸고 더한다.

빼는 함수: 두 분모를 둘의 최소 공배수로 바꾸고 뺀다.

곱하는 함수: 분자끼리, 분모끼리 곱한다.

나누는 함수: 앞 분자와 뒤 분모를 곱하고, 앞 분모와 뒤 분자를 곱한다.

**1.3 소스 코드**

|  |
| --- |
| #include <iostream>  using namespace std;  struct bunsu{  int ja, mo;  };  //분수 구조체  int add(int ja1, int mo1, int ja2, int mo2);//더하는 함수  int sub(int ja1, int mo1, int ja2, int mo2);//빼는 함수  int multi(int ja1, int mo1, int ja2, int mo2);//곱하는 함수  int div(int ja1, int mo1, int ja2, int mo2);//나누는 함수  int yak(int x, int y);//최대공약수를 구하는 함수  int bae(int x, int y);//최소공배수를 구하는 함수  int main() {  struct bunsu su1;  struct bunsu su2;  char gyaesan[5];  cin >> su1.ja >> su1.mo;  cin >> gyaesan;  cin >> su2.ja >> su2.mo;  if(gyaesan[0] == 'a')  add(su1.ja, su1.mo, su2.ja, su2.mo);  else if(gyaesan[0] == 's')  sub(su1.ja, su1.mo, su2.ja, su2.mo);  else if(gyaesan[0] == 'm')  multi(su1.ja, su1.mo, su2.ja, su2.mo);  else if(gyaesan[0] == 'd')  div(su1.ja, su1.mo, su2.ja, su2.mo);  return 0;  }  /\*유클리드 호제법  1. x를 y로 나눈 나머지를 구한다.  2. y를 x로, 나머지를 y로 놓고 1.의 과정을 반복한다.  3. 나머지가 0이 될 때까지 2.과정을 반복한다.  4. 나머지가 0이 됐을 때 y값이 최대공약수이다.  5. 최소공배수는 최초의 x\*y/(최대공약수)이다.  \*/  int yak(int x, int y)//최대공약수를 구하는 함수  {  int n;  while(1)  {  n = x % y;  if(n == 0)  return y;  x = y;  y = n;  }  }  int bae(int x, int y)//최소 공배수를 구하는 함수  {  return x \* y / yak(x, y);  }  int add(int ja1, int mo1, int ja2, int mo2)  {  int n1 = bae(mo1, mo2)/mo1;  int n2 = bae(mo1, mo2)/mo2;  //두 분모가 각각 최소공배수가 되기 위해 곱해야 하는 수  int add\_ja;  add\_ja = (ja1 \* n1) + (ja2 \* n2);  if(1 == bae(mo1, mo2) / yak(add\_ja, bae(mo1, mo2)))//만약 기약분수 형태를 취한 분모가 1이면  cout << add\_ja / yak(add\_ja, bae(mo1, mo2));//기약분수 형태를 취한 분자 출력  else if(add\_ja == 0)  cout << "0";//답이 0이면 0출력  else  cout << add\_ja / yak(add\_ja, bae(mo1, mo2)) << "/" << bae(mo1, mo2) / yak(add\_ja, bae(mo1, mo2));  //기약 분수 형태로 출력하기위해 분자와 분모를 각각 둘의 최대공약수로 나눔  return 0;  }  int sub(int ja1, int mo1, int ja2, int mo2)  {  int n1 = bae(mo1, mo2)/mo1;  int n2 = bae(mo1, mo2)/mo2;  //두 분모가 각각 최소공배수가 되기 위해 곱해야 하는 수  int sub\_ja;  sub\_ja = (ja1 \* n1) - (ja2 \* n2);  if(1 == bae(mo1, mo2) / yak(sub\_ja, bae(mo1, mo2)))//만약 기약분수 형태를 취한 분모가 1이면  cout << sub\_ja / yak(sub\_ja, bae(mo1, mo2));//기약분수 형태를 취한 분자 출력  else if(sub\_ja == 0)  cout << "0";  else  cout << sub\_ja / yak(sub\_ja, bae(mo1, mo2)) << "/" << bae(mo1, mo2) / yak(sub\_ja, bae(mo1, mo2));  //기약 분수 형태로 출력하기위해 분자와 분모를 각각 둘의 최대공약수로 나눔  return 0;  }  int multi(int ja1, int mo1, int ja2, int mo2)  {  int multi\_ja, multi\_mo;  multi\_ja = ja1 \* ja2;//두 분수를 곱한 분자  multi\_mo = mo1 \* mo2;//두 분수를 곱한 분모  if(1 == multi\_mo / yak(multi\_ja, multi\_mo))//만약 기약분수 형태를 취한 분모가 1이면  cout << multi\_ja / yak(multi\_ja, multi\_mo);//기약분수 형태를 취한 분자 출력  else  cout << multi\_ja / yak(multi\_ja, multi\_mo) << "/" << multi\_mo / yak(multi\_ja, multi\_mo);  //기약분수 형태로 출력하기 위해 분자와 분모의 최대공약수로 각각 나눈 값을 출력  return 0;  }  int div(int ja1, int mo1, int ja2, int mo2)  {  int div\_ja, div\_mo;  div\_ja = ja1 \* mo2;//앞 분수 분자 \* 뒤 분수 분모  div\_mo = mo1 \* ja2;//앞 분수 분모 \* 뒤 분수 분자  if(1 == div\_mo / yak(div\_ja, div\_mo))//기약분수 형태를 취한 분모가 1이면  cout << div\_ja / yak(div\_ja, div\_mo);//기약분수 혀태를 취한 분자 출력  else  cout << div\_ja / yak(div\_ja, div\_mo) << "/" << div\_mo / yak(div\_ja, div\_mo);  //기약분수 형태로 출력하기 위해 분자와 분모의 최대공약수로 각각 나눈 값을 출력  return 0;  } |

**1.4 결과 화면**

|  |
| --- |
|  |

**2. 특정 조건까지 계속 입력 받는 프로그램**

**2.1 문제 정의**

-> 동적 할당을 이용해 특정 수가 입력될 때까지 입력된 수들의 총합을 구한다.

**2.2 문제 해결 방법**

-> 특정 수가 입력될 때까지 배열의 새로운 칸을 만들고 입력된 수를 총합에 더하는 것을 반복한다.

**2.3 소스 코드**

|  |
| --- |
| #include <iostream>  using namespace std;  int main()  {  int \*a;  int x = 0;  int chonghop = 0;    while(1)  {  a = new int[x+1];//반복문을 실행할 때 마다 배열의 다음 칸을 만듦    cin >> a[x];//입력    if(a[x] == -2147483648)  break;    chonghop += a[x];//지금까지의 합에 더함    x++;  }  cout << chonghop << endl;    delete[] a;//반환  return 0;  } |

**2.4 결과 화면**

|  |
| --- |
|  |

**3. 학생 수 만큼 구조체를 입력 받는 프로그램**

**3.1 문제 정의**

-> 학생 수를 입력 받고 그 수에 따라 구조체를 입력 받음

**3.2 문제 해결 방법**

-> 입력 받은 학생 수를 이용해 구조체 수를 동적 할당하고 이름, 성적을 입력 받는다. 그 후 abcd기준에 따라 abcd를 정한다.

**3.3 소스 코드**

|  |
| --- |
| #include <iostream>  using namespace std;  struct student  {  char\* name;  int math[2];  char mathGrade;  int eng[2];  char engGrade;  };  int main()  {  int number, i;  cin >> number;  struct student\* s;  s = new struct student[number];//동적할당으로 학생 구조체를 만든다  for(i=0; i<number; i++)  {  s[i].name = new char[100];//동적할당으로 학생의 이름을 만든다  cin >> s[i].name;  cin >> s[i].math[0] >> s[i].math[1];  cin >> s[i].eng[0] >> s[i].eng[1];  }    for(i=0; i<number; i++)//ABCD의 기준에 따라 mathGrade, engGrade를 설정  {  if(s[i].math[0] + s[i].math[1] >= 180)  s[i].mathGrade = 'A';  else if(s[i].math[0] + s[i].math[1] <= 179 && s[i].math[0] + s[i].math[1] >= 160)  s[i].mathGrade = 'B';  else if(s[i].math[0] + s[i].math[1] <= 159 && s[i].math[0] + s[i].math[1] >= 130)  s[i].mathGrade = 'C';  else if(s[i].math[0] + s[i].math[1] <= 129 && s[i].math[0] + s[i].math[1] >= 100)  s[i].mathGrade = 'D';  else  s[i].mathGrade = 'F';    if(s[i].eng[0] + s[i].eng[1] >= 180)  s[i].engGrade = 'A';  else if(s[i].eng[0] + s[i].eng[1] <= 179 && s[i].eng[0] + s[i].eng[1] >= 160)  s[i].engGrade = 'B';  else if(s[i].eng[0] + s[i].eng[1] <= 159 && s[i].eng[0] + s[i].eng[1] >= 130)  s[i].engGrade = 'C';  else if(s[i].eng[0] + s[i].eng[1] <= 129 && s[i].eng[0] + s[i].eng[1] >= 100)  s[i].engGrade = 'D';  else  s[i].engGrade = 'F';  }    for(i=0; i<number; i++)  cout << s[i].name << " " << s[i].mathGrade << " " << s[i].engGrade << endl;  //출력  delete[] s;  return 0;  } |

**3.4 결과 화면**

|  |
| --- |
|  |

**4. 특정 조건까지 구조체를 입력 받는 프로그램**

* 1. **문제 정의**

-> null이 입력될 때 까지 학생 구조체를 입력 받는 프로그램

**4.2 문제 해결 방법**

-> 최초 동적할당 배열을 10000만큼 잡고 null이 입력될 때 까지 인원수를 세서 활용한다.

**4.3 소스 코드**

|  |
| --- |
| #include <iostream>  using namespace std;  struct student  {  char\* name;  int math[2];  char mathGrade;  int eng[2];  char engGrade;  };  int main()  {  int x=0;  struct student\* s;  s = new struct student[10000];//동적할당으로 학생 구조체를 만든다    while(1)  {  s[x].name = new char[100];//동적할당으로 학생의 이름을 만든다  cin >> s[x].name;  if(s[x].name[0] == 'n' && s[x].name[1] == 'u' && s[x].name[2] == 'l' && s[x].name[3] == 'l')  break;//null이 입력되면 break  cin >> s[x].math[0] >> s[x].math[1];  cin >> s[x].eng[0] >> s[x].eng[1];    x++;//인원 수 추가  }    int i;  for(i=0; i<x; i++)//ABCD의 기준에 따라 mathGrade, engGrade를 설정  {  if(s[i].math[0] + s[i].math[1] >= 180)  s[i].mathGrade = 'A';  else if(s[i].math[0] + s[i].math[1] <= 179 && s[i].math[0] + s[i].math[1] >= 160)  s[i].mathGrade = 'B';  else if(s[i].math[0] + s[i].math[1] <= 159 && s[i].math[0] + s[i].math[1] >= 130)  s[i].mathGrade = 'C';  else if(s[i].math[0] + s[i].math[1] <= 129 && s[i].math[0] + s[i].math[1] >= 100)  s[i].mathGrade = 'D';  else  s[i].mathGrade = 'F';    if(s[i].eng[0] + s[i].eng[1] >= 180)  s[i].engGrade = 'A';  else if(s[i].eng[0] + s[i].eng[1] <= 179 && s[i].eng[0] + s[i].eng[1] >= 160)  s[i].engGrade = 'B';  else if(s[i].eng[0] + s[i].eng[1] <= 159 && s[i].eng[0] + s[i].eng[1] >= 130)  s[i].engGrade = 'C';  else if(s[i].eng[0] + s[i].eng[1] <= 129 && s[i].eng[0] + s[i].eng[1] >= 100)  s[i].engGrade = 'D';  else  s[i].engGrade = 'F';  }    for(i=0; i<x; i++)  cout << s[i].name << " " << s[i].mathGrade << " " << s[i].engGrade << endl;  //출력    delete[] s;  return 0;  } |

**4.4 결과 화면**

|  |
| --- |
|  |