<알고리즘 과제 #1>

진수변환 알고리즘

과목명: 컴퓨터 알고리즘(01분반)

담당교수: 김명주 교수

소속학과: 정보보호학과

학번: 2020111318

이름: 김수현

목 차

1. 32비트 정수의 표현범위

컴퓨터의 처리 단위
4bit
8bit
16bit
32bit
64bit
128bit

대표적으로 컴퓨터는 위의 표와 같이 bit별로 데이터를 처리할 수 있다. 그 중에서 컴퓨터의 저장 단위로 가장 대표적인 "32bit(4byte)" 시스템을 중점적으로 다룰 것이다.

기본적으로 컴퓨터는 이진수(0,1)만을 사용할 수 있다. 이에 따라, 데이터의 단위가 2^n로 이루어져 있다.

따라서, 32bit인 경우에는 2^32의 데이터 크기를 사용할 수 있게 되는 것이다.

하지만, 수의 체계에는 양의 정수 뿐만 아니라 음의 정수도 존재한다. 만약, 양의 정수만을 고려한다면 0~2^32 인 것이다. 이 범위는 음의 정수를 표현할 수 없으며 이에 따라 음수를 표현하기 위한 방법으로 "2의 보수" 를 이용한다.

또한, 양의 정수와 음의 정수를 구분하기 위해서 가장 최상위 비트를 부호비트로 사용한다. (0 : 양의 정수, 1: 음의 정수)

보통 n개의 bit가 있으면 2^n개의 수를 표현할 수 있는데, 0부터 시작하므로 최대 표현 가능한 수는 2^n – 1 이 된다. 그리고 부호가 있기 때문에 음수 범위로 절반이 이동함으로써 32bit의 수의 표현 범위는 -2^31(-2,147,483,648) ~ 2^31 – 1 (2,147,483,647) 이다.

2. 진수변환 문제정의

사용자로부터 양의 정수 또는 음의 정수를 입력 받아 2진수, 8진수, 16진수로 변환하여 모두 출력해줘야 한다.

모든 경우는 32bit인 경우를 한정시키기 때문에, 32자리의 0과 1로 이루어진 이진수로 표현할 수 있다.

1) 2진수 표현방법

-> 먼저, 양의 정수이면 그대로 사용하지만 음의 정수라면 음수부호(-)를 생략한다.

그 다음, 2진수로 변환을 해준다음 1의 보수화를 해준다. 마지막으로 +1을 해주면

우리가 입력한 정수 값이 2진수의 형태로 변환이 될 수 있다.

2) 8진수 표현방법

-> 일단은 2진수 형태로 표현을 해준다음 8진수로 변환해주는 것이 가장 효율적이다. 그런 다음, 각각의 자리를 3개씩 나누고 1이 있는 자리만 각 자리에 맞는 2의 누승을 곱해줘서 합쳐준다. (더하는 것이 아니라 합치는 것이다. 예를 들어, 001 011 -> 13)

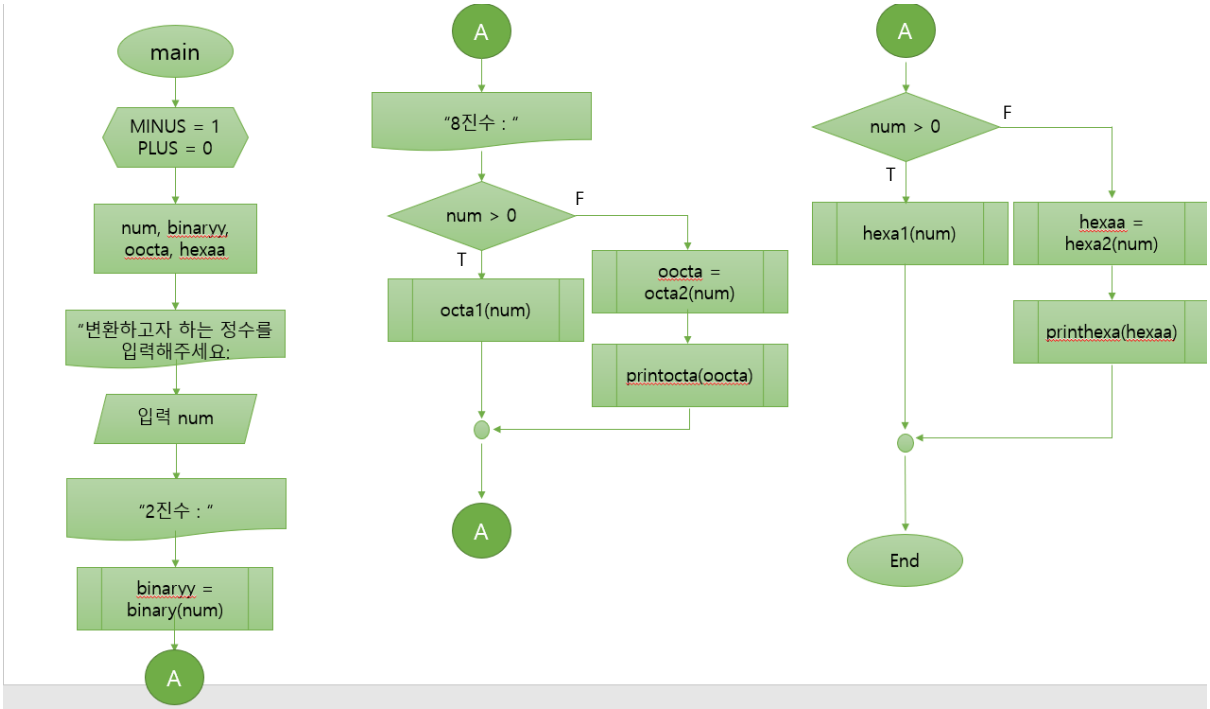
3_ 16진수 표현방법

-> 16진수도 8진수와 비슷한 방법으로 먼저 2진수 형태로 표현해준 다음, 각각의 자리를 4개씩 나누어 1이 있는 자리만 각 자리에 맞는 2의 누승을 곱해줘서 이것을 다시 16진수로 변환시켜준다.

3. 프로그램 설계(순서도와 설명)

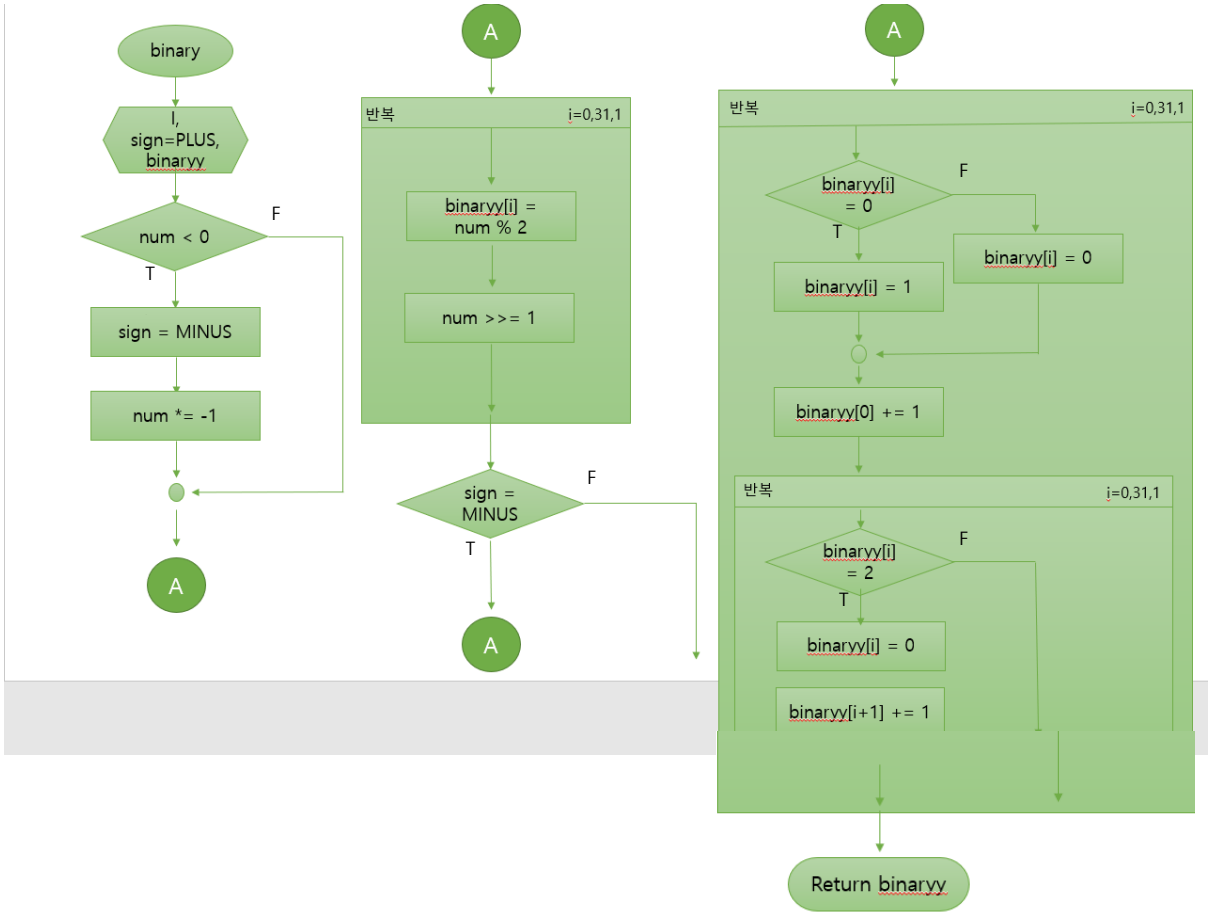
앞에서 진수변환에 대한 정확한 이해와 각 진수 별 변환방법을 알아보았다. 그 다음, 알고리즘에서 가장 중요한 부분인 '순서도 작성'과 이에 대한 부가적인 설명으로 실제로 코드로 구현하기 전 준비단계를 완벽히 마쳐야 한다.

<main 함수 순서도>



전반적인 main 함수 순서도 흐름을 설명해보자면, 사용자로부터 정수 값을 입력 받은 후 2진수를 출력하려면 binary함수를 호출하고, 8진수와 16진수 같은 경우에는 입력 받은 정수가 양의 정수라면, octa1 과 hexa1을 호출하고 만약 음의 정수라면 octa2와 hexa2 함수를 호출하여 출력한다.

<2진수 변환 함수 순서도>

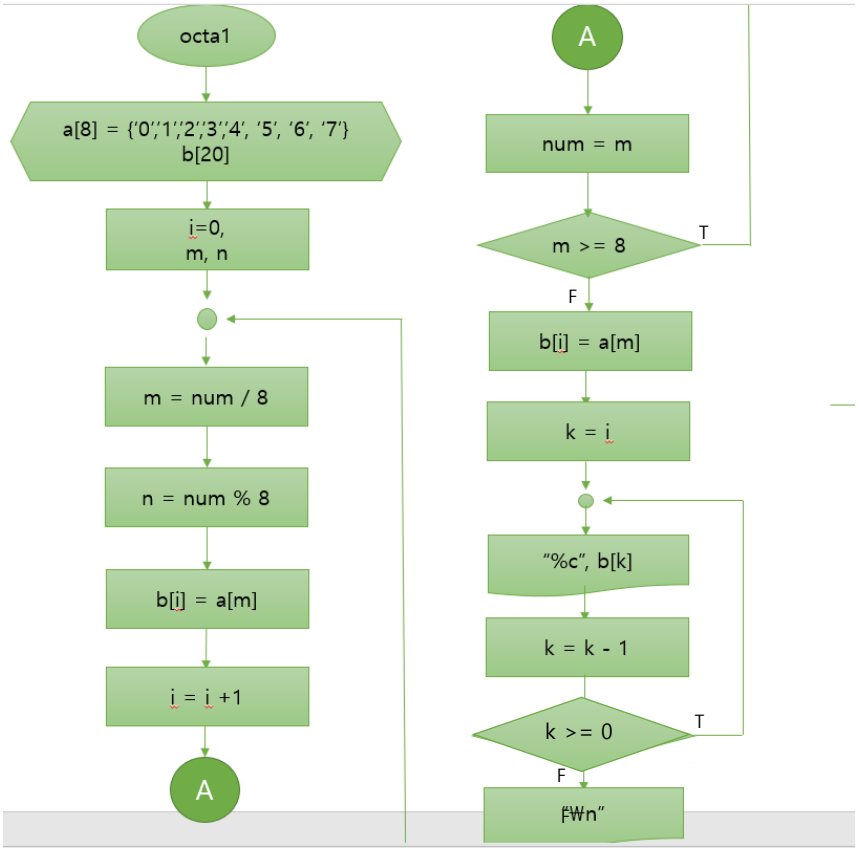


2진수 변환 시에는 입력 받은 정수가 음수라면, 최상위 비트 부호가 MINUS이고 마이너스 부호를 제거하기 위해 -1을 곱해준다. 그 다음, 32비트를 표현해주기 위해 32번 반복하여 입력 받은 정수를 2로 나눈 나머지를 binaryy 배열변수에 차례대로 넣고, 오른쪽으로 1씩 비트이동을 해준다.

만약, 입력 받은 정수가 음의 정수라면, 1의 보수 방법을 적용시켜줘야 되기 때문에 각 자리 비트가 0이면 1로, 1이면 0으로 바꿔주는 연산을 한다. 그런 다음, 1을 더해준다. 또한, 캐리가 발생하는 경우도 고려를 해줘야 되기 때문에 binaryy[i] = 2라면 binaryy[i] = 0, binaryy[i+1] += 1의 연산을 시행해준다. 이렇게 다 끝난 후에는 각 비트를 담아놓았던 배열 변수 binaryy를 return 해준다.

< 8진수 변환 함수 순서도>

1. 양의 정수 일 때,



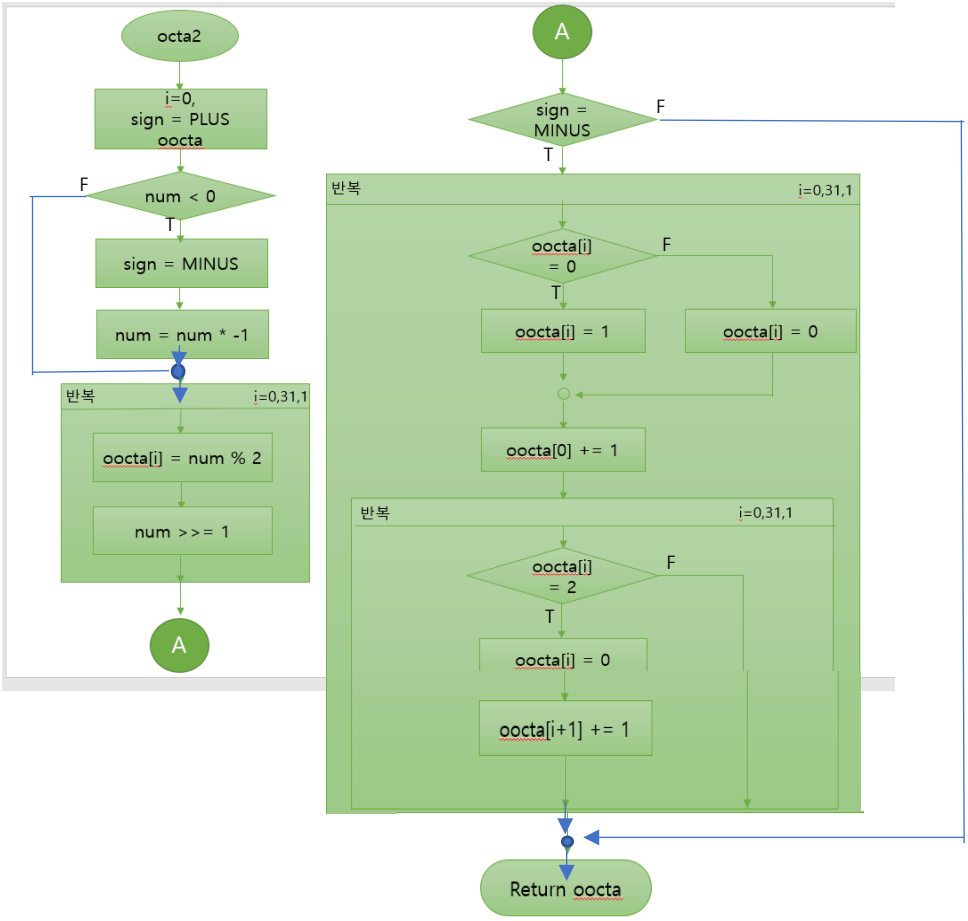
a[8]이라는 배열 변수를 선언해주는데, 이 때 8진수를 이루고 있는 0~7 사이의 수를 char타입으로 담아준다. 그 다음, 인덱스 변수 i와 num을 8로 나눈 몫 변수 m, num을 8로 나눈 나머지 변수 n을 선언해준다.

그리고, num을 8로 나눈 몫 변수를 인덱스로 활용하여 a[]배열을 i를 인덱스변수로 갖는 배열 b에 대입해준다. 그런 다음, i의 값을 하나씩 증가시킨다.

8로 나눈 몫 변수 m을 새로운 num의 값으로 업데이트 시켜주고, m이 8보다 큰 경우에만 위의 모든 연산과정들을 실행시켜준다.

만약, 몫 변수 m이 8보다 작다면, m을 인덱스 값으로 한 배열 a의 값들을 b[i]에 업데이트 시켜준다. 그리고 i를 변수 k에 넣는다. 그리고 나서, k가 0보다 클 경우에만 b[k]에 값을 집어넣는데, k의 값은 반복을 하면서 하나씩 줄어든다.

2. 음의 정수 일 때,



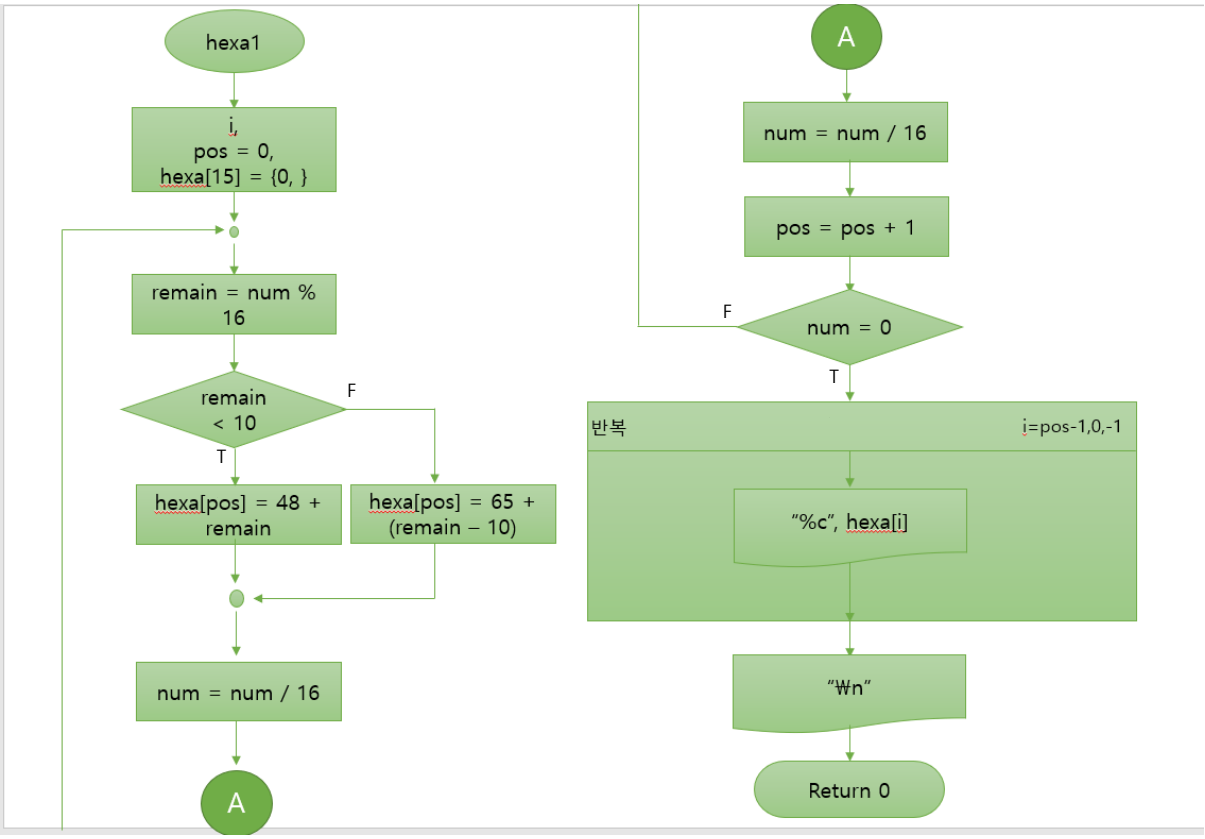
음의 정수를 8진수로 바꾸는 알고리즘은 음의 정수를 2진수로 변환하는 과정과 똑같다. 다만, 배열 변수를 8진수를 나타내는 oocta 변수를 사용해준다.

2진수 변환과정과 똑같은 이유는 컴퓨터는 음의 정수를 표현해줄 수 없기 때문에, 8진수든 16진수든 2진수가 아닌 진수로 변환하기 위해서는 무조건 2의 보수법을 활용하여 변환시켜줘야 한다.

따라서, 음의 정수를 8진수로 변환시켜 주기 위해서는 2의 보수법을 활용한 2진수로 바꿔주고 그런 다음에 8진수로 바꿔주는 구현 방식을 생각해야 한다.

<16진수 변환 함수 순서도>

1) 양의 정수 일 때



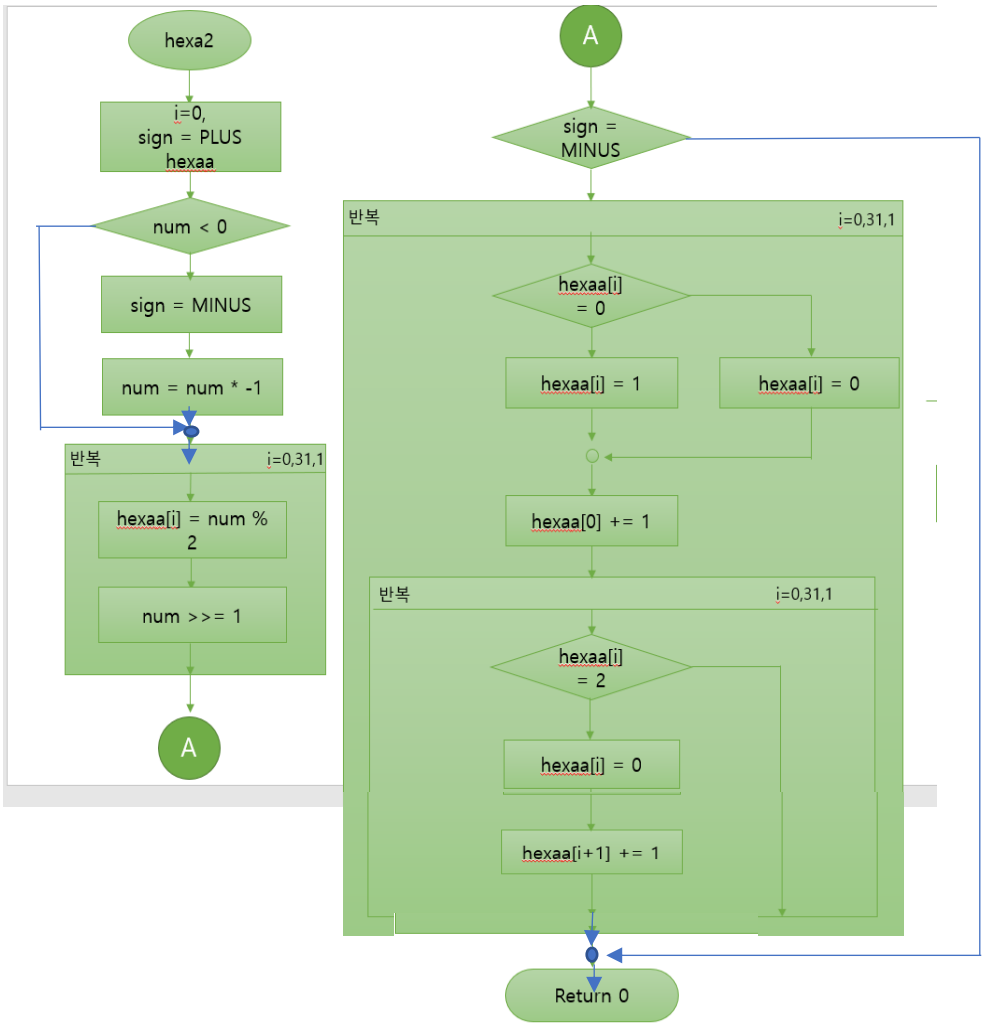
인덱스 변수 I, 위치변수 pos(초기화된 상태)와 hexa라는 이름을 가진 충분한 크기의 배열변수를 선언해준다.

그 후, 입력 받은 양의 정수를 16으로 나눠준 나머지 값을 remain이라는 변수에 넣어주고, 그 값이 10보다 작다면, hexa[pos]에 48과 remain값을 더해준 값을 넣는다. 만약 그렇지 않다면, 65에 remain-10을 더해준 값을 넣는다.(이 과정은 아스키코드를 고려하기 위한 과정이다.)

그리고나서, num을 16으로 나눈 값을 새로운 num값으로 업데이트 시켜준다. (pos는 1씩 증가)

num이 0이 아닐 때까지 반복하다가, 0이 되면 역순으로 hexa[i]의 값을 char타입으로 출력해준다.

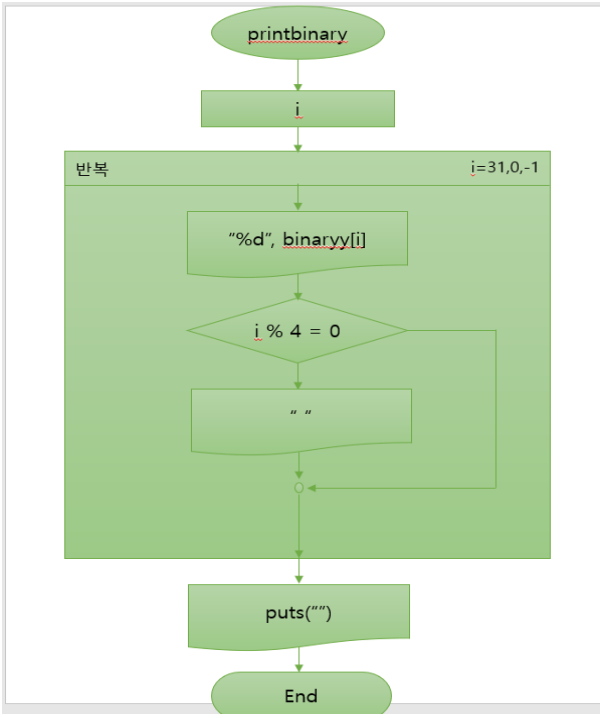
2) 음의 정수 일 때



입력 받은 음의 정수를 16진수로 변환하는 것 또한 음의 정수를 2진수로 변환하는 알고리즘과 똑같다. 8진수에서도 설명했듯이, 컴퓨터가 음의 정수를 표현할 수 없기 때문에 2의 보수법을 취하고 그 후에 16진수로 변환시켜줘야 되기 때문에 일단은 2진수로 변환시켜줘야 한다.

<print 함수(2진수, 8진수, 16진수)>

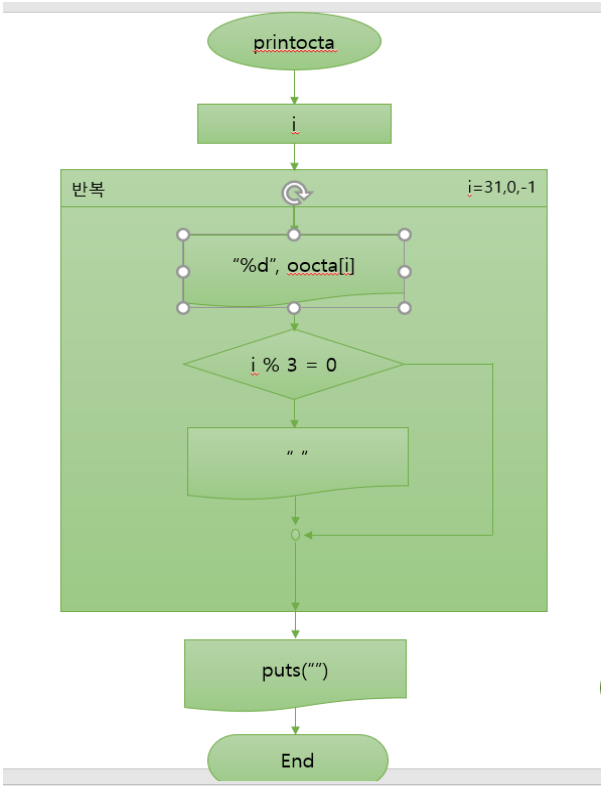
1) 2진수



앞에 binary함수에서 return받은 binaryy값을 출력하기 위해서 printbinary함수를 따로 만들어준다.

역순으로 binaryy[i]의 값을 출력하는데, 4자리씩 끊어서 출력시켜 주기 위해서 i % 4 = 0 연산을 실행시켜준다.

2) 8진수

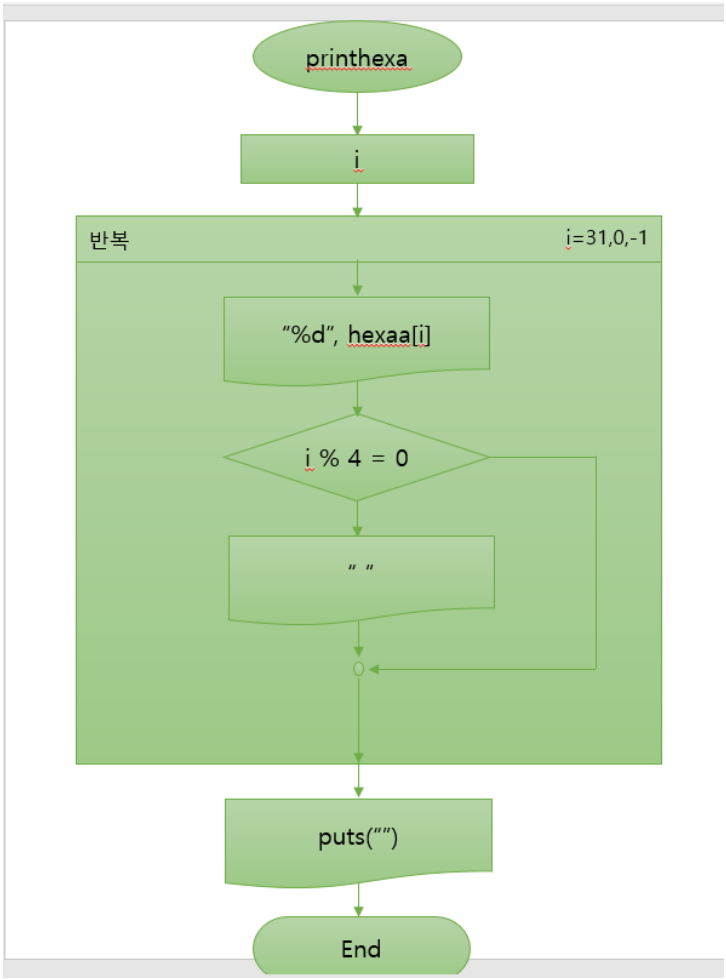


octa2 함수에서 return받은 oocta값을 출력시켜 주기 위해서 printocta함수를 따로 만들어준다.

2진수와는 달리 8진수는 3자리씩 나눠서 출력을 해줘야 하기 때문에 i % 3 = 0의 연산을 해준다.

(사실 2진수를 3자리씩 나눠서 이를 8진수로 변환을 해줘야 하는데 이 과정을 수행하지 못했다.)

3) 16진수



hexa2 함수에서 return받은 hexaa 값을 출력시켜 주기 위해서 printhexa함수를 따로 만들어준다.

2진수와는 달리 16진수는 4자리씩 나눠서 출력을 해줘야 하기 때문에 i % 4 = 0의 연산을 해준다.

(사실 2진수를 4자리씩 나눠서 이를 16진수로 변환을 해줘야 하는데 이 과정을 수행하지 못했다.)

<프로그램 소스>

```c
#include <stdio.h>
#include <string.h>

#define MINUS 1
#define PLUS 0 // 부호판별 위해 최상위비트가  0 이면 양의 정수, 1이면 음의 정수 라고 정의함.


// 1. 입력 받은 정수를 2진수로 변환하기(양의 정수, 음의 정수 포함)
int* binary(int num) {

    int i;
    int sign = PLUS;
    int* binaryy = (int*)malloc(sizeof(int) * 32); // binaryy라는 이름의 동적메모리 할당.

    if (num < 0) {
        sign = MINUS;
        num = num * -1;

    } // 입력받은 정수가 음의 정수라면, 부호비트는 MINUS로
    // 입력받은 음의 정수는 -1을 곱함으로써 양의 정수로 바꿔준다.

    for (i = 0; i < 32; i++) {
        binaryy[i] = num % 2; // 입력받은 정수를 2로 나눈 나머지를 binaryy 배열 변수에 차례대로 담는다.
        num >>= 1; // 각 비트를 오른쪽으로 1씩 옮김.
    }


    if (sign == MINUS) { // 입력받은 정수가 음의 정수라면
        for (i = 0; i < 32; ++i) {
            if (binaryy[i] == 0) {
                binaryy[i] = 1;
            }
            else
                binaryy[i] = 0;
        } // 각 자릿수를 비교하여 0이면 1로, 1이면 0으로 변환시켜준다. (1의보수 변환방법)

        binaryy[0] += 1; // 1을 더해준다.

        for (i = 0; i < 32; ++i) {
            if (binaryy[i] == 2) {
                binaryy[i] = 0;
                binaryy[i + 1] += 1; // 1을 더해주면 캐리가 발생하기 때문에, 캐리가 발생하는 것도 처리해준다.
            }
            else
            {
                break;
            }
        }
    }
    return binaryy;
}

// 2. 입력받은 정수(양의 정수)를 8진수로 변환
void octa1(int num) {

    int a[8] = { '0', '1', '2', '3', '4', '5', '6', '7' }; // 8진수 배열 a[8] 선언
    int b[20];
    int i = 0;
    int m, n;

    do {
        m = num / 8; // 입력받은 정수를 8로 나눴을때의 몫을 m에 저장.
        n = num % 8; // 입력받은 정수를 8로 나눴을때의 나머지를 n에 저장.
        b[i] = a[n];
        i++;
        num = m;

    } while (m >= 8); // 몫이 8포함하여 클때까지 반복

    b[i] = a[m];
    int k = i;

    do {
        printf("%c", b[k]);
        k--;
    } while (k >= 0);

```

```
77      printf("₩n");
78  }
79
80
81
82
83
84  // 3. 입력받은 정수(음의 정수)를 8진수로 변환하기
85  int* octa2(int num) {
86
87      int i;
88      int sign = PLUS;
89      int* oocta= (int*)malloc(sizeof(int) * 32); // oocta라는 이름의 동적메모리 할당
90
91      if (num < 0) {
92          sign = MINUS;
93          num = num * -1;
94
95      } // 입력받은 정수가 음의 정수라면, 부호비트는 MINUS로
96      // 입력받은 음의 정수는 -1을 곱함으로써 양의 정수로 바꿔준다.
97
98      for (i = 0; i < 32; i++) {
99          oocta[i] = num % 2; // 입력받은 정수를 2로 나눈 나머지를 binaryy 배열 변수에 차례대로 담는다.
100         num >>= 1; // 각 비트를 오른쪽으로 1씩 옮김.
101     }
102
103     if (sign == MINUS) {
104         for (i = 0; i < 32; ++i) {
105             if (oocta[i] == 0) { // 입력받은 정수가 음의 정수라면
106                 oocta[i] = 1;
107             }
108             else
109                 oocta[i] = 0;
110         } // 각 자릿수를 비교하여 0이면 1로, 1이면 0으로 변환시켜준다. (1의보수 변환방법)
111
112         oocta[0] += 1; // 1을 더해준다.
113
114         for (i = 0; i < 32; ++i) {
115             if (oocta[i] == 2) {
116                 oocta[i] = 0;
117                 oocta[i + 1] += 1;
118             } // 1을 더해주면 캐리가 발생하기 때문에, 캐리가 발생하는 것도 처리해준다.
119             else
120             {
121                 break;
122             }
123         }
124
125     }
126     return oocta;
130 // 4. 입력받은 정수(양의 정수)를 16진수로 변환하기
131 void hexa1(int num) {
132
133
134     int i;
135     int pos = 0;
136     char hexa[15] = { 0, }; // 충분한 크기의 hexa배열을 선언한다.(char 타입)
137
138     while (1)
139     {
140         int remain = num % 16;    // 16으로 나누었을 때의 나머지를 구한다.
141
142         if (remain < 10)
143         {
144             hexa[pos] = 48 + remain;
145         } // 나머지가 10보다 작다면, 숫자 0의 아스키코드 값 48에 나머지를 더한것을 hexa[pos]에 저장한다.
146
147         else
148         {
149             hexa[pos] = 65 + (remain - 10);
150         } // 나머지가 10보다 크거나 같다면, 대문자 A의 아스키코드값에 65를 더한 것과 나머지에서 10을 뺀 값 두개를 더한다.
151
152         num = num / 16; // 16으로 나눈 몫을 저장한다.
153
154         pos++; //
```

```
        if (num == 0)    // 입력받은 정수가 0이라면 break
            break;
    }


    for (i = pos - 1; i >= 0; i--)
    {
        printf("%c", hexa[i]);
    } // hexa 배열의 요소들을 역순으로 출력한다.

    printf("₩n");

    return 0;

}

// 5. 입력받은 음의 정수를 16진수로 변환
int* hexa2(int num) {

    int i;
    int sign = PLUS;
    int* hexaa = (int*)malloc(sizeof(int) * 32); // hexaa라는 이름의 동적메모리 할당

    if (num < 0) {
        sign = MINUS;
        num *= -1;

    } // 입력받은 정수가 음의 정수라면, 부호비트는 MINUS로
    // 입력받은 음의 정수는 -1을 곱함으로써 양의 정수로 바꿔준다.

    for (i = 0; i < 32; i++) {
        hexaa[i] = num % 2; // 입력받은 정수를 2로 나눈 나머지를 binaryy 배열 변수에 차례대로 담는다.
        num >>= 1; // 각 비트를 오른쪽으로 1씩 옮김.
    }

    if (sign == MINUS) { // 입력받은 정수가 음의 정수라면
        for (i = 0; i < 32; ++i) {
            if (hexaa[i] == 0) {
                hexaa[i] = 1;
            }
            else
                hexaa[i] = 0;
        } // 각 자릿수를 비교하여 0이면 1로, 1이면 0으로 변환시켜준다. (1의보수 변환방법)

        hexaa[0] += 1; // 1을 더해준다.

        for (i = 0; i < 32; ++i) {
            if (hexaa[i] == 2) {
                hexaa[i] = 0;
                hexaa[i + 1] += 1;
            } // 1을 더해주며 캐리가 발생하기 때문에  캐리가 발생하는 것도 처리해준다.
            else
            {
                break;
            }
        }

    }


    return hexaa;
}


// 6. 입력받은 정수 -> 2진수 출력함수
void printbinary(int* binaryy) {

    int i;
    for (i = 31; i >= 0; --i) {
        printf("%d", binaryy[i]); // 역순으로 bianryy 배열 요소들을 출력한다.
        if (i % 4 == 0) {
            printf(" ");
        } // 4자리씩 끊기 위한 처리.
    }
    puts("");
}
```

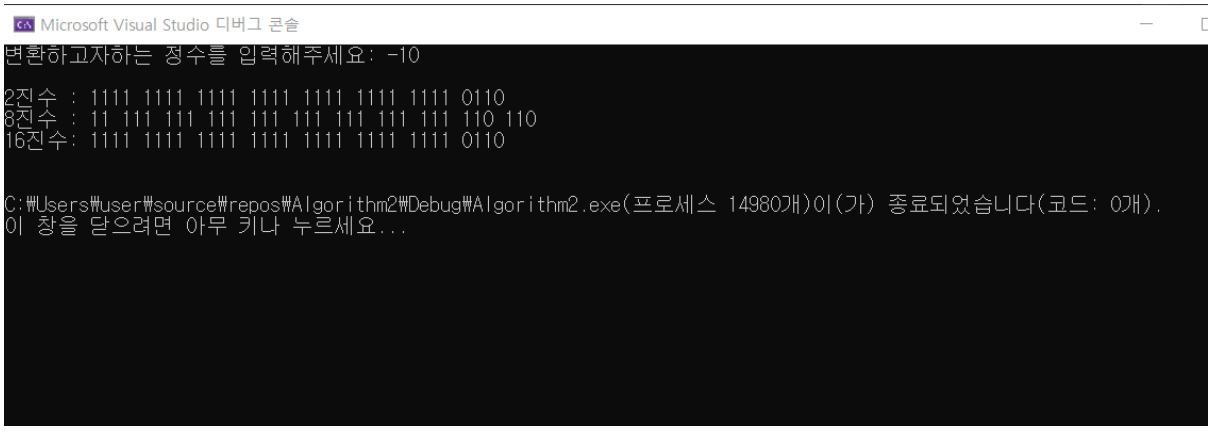
```c
// 6. 입력받은 정수 -> 2진수 출력함수
void printbinary(int* binaryy) {

    int i;
    for (i = 31; i >= 0; --i) {
        printf("%d", binaryy[i]); // 역순으로 bianryy 배열 요소들을 출력한다.
        if (i % 4 == 0) {
            printf(" ");
        } // 4자리씩 끊기 위한 처리.
    }
    puts("");
}

// 7. 입력받은 정수 -> 8진수 출력함수
void printocta(int* oocta) {

    int i;
    for (i = 31; i >= 0; --i) {
        printf("%d", oocta[i]); // 역순으로 oocta 배열 요소들을 출력한다.
        if (i % 3 == 0) {
            printf(" ");
        } // 3자리씩 끊기 위한 처리.
    }
    puts("");

// 8. 입력받은 정수 ->16진수 출력함수
void printhexa(int* hexaa) {

    int i;
    for (i = 31; i >= 0; --i) {
        printf("%d", hexaa[i]); // 역순으로 hexaa 배열 요소들을 출력한다.
        if (i % 4 == 0) {
            printf(" ");
        } // 4자리씩 끊기 위한 처리.
    }
    puts("");

}


// main() 함수
int main() {

    int num;
    int* binaryy;
    int* oocta;
    int* hexaa; // binaryy(2진수), oocta(음의 정수일때 8진수 변환), hexaa(음의 정수일때 16진수 변환) 포인터 선언.

    printf("변환하고자하는 정수를 입력해주세요: ");
    scanf("%d", &num); // 입력받은 정수 입력
    printf("₩n");


    printf("2진수 : ");

    binaryy = binary(num);
    printbinary(binaryy); // 2진수 출력

    printf("8진수 : ");

    if (num > 0) {
        octa1(num);
    } // 양의 정수일 때는 octa1함수를 호출
    else {
        oocta = octa2(num);
        printocta(oocta);
    } // 음의 정수일 때는 octa2함수를 호출

    printf("16진수: ");

    if (num > 0) {
        hexa1(num);
    } // 양의 정수일 때는 hexa1함수를 호출
    else {
        hexaa = hexa2(num);
```

```
297         printhexa(hexaa);
298     } // 음의 정수일 때는 hexa2함수를 호출
299
300
301     printf("₩n");
302
303
304     return 0;
305
306
307 }
308
```

<프로그램 실행화면>

1. 입력정수가 양의 정수 일 때

2. 입력정수가 음의 정수 일 때



(2진수는 변환에 성공했지만, 8 & 16진수는 끝내 하지 못했다.)