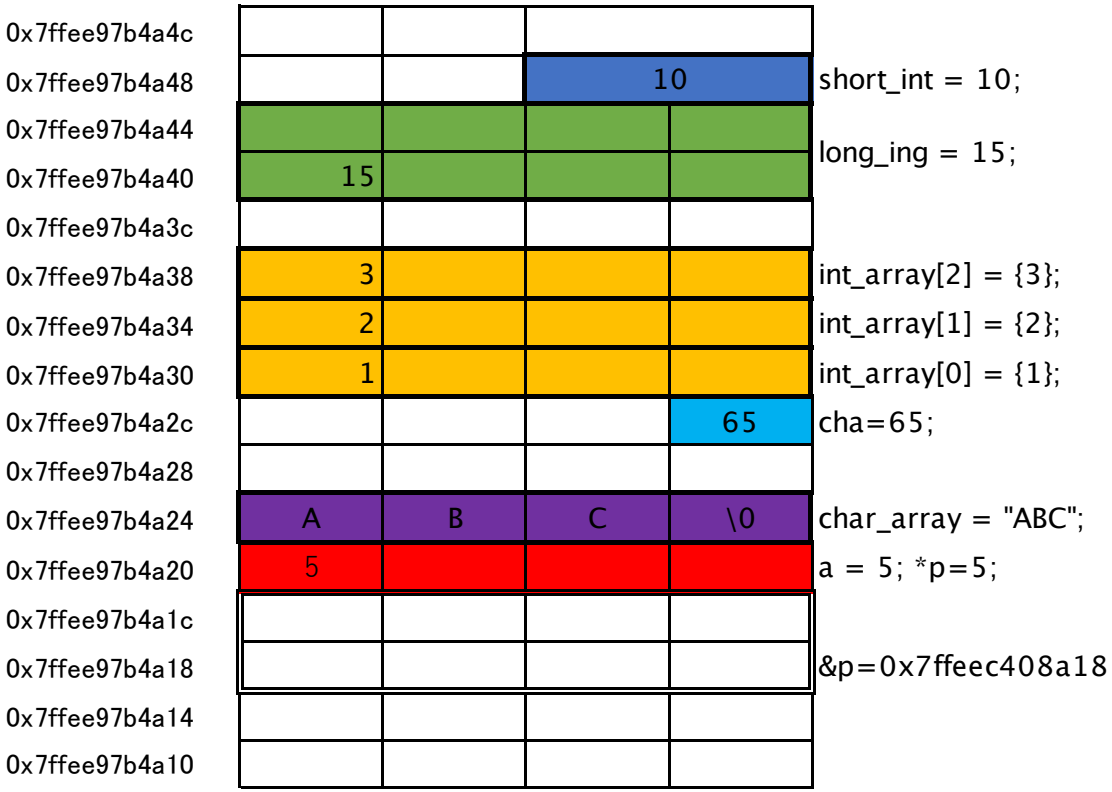
最初に，整数型の記憶領域の大きさが異なる二種類の整数型変数と値,アドレスについて説明する．変数 short_int の型は short であり，変数 long_int の型は long である. sizeof()により short のサイズが 2byte, long のサイズが 8byte であることを確認した．変数 short_int=10 の address は 0x7ffee97b4a4a であり，青色で示した示した領域(0x7ffee97b4a4a～0x7ffee97b4a4b)が変数 short_int の格納場所となる．変数 long_int=15 の address は 0x7ffee97b4a40 であり，緑色で示した領域(0x7ffee97b4a40～0x7ffee97b4a48)が変数 long_int の格納場所となる.

次に，整数型の配列変数とアドレスについて説明する.
int_array[3]={1,2,3};と初期化を行い, for 文で各値とアドレスを取り出した. sizeof()により int のサイズは 4byte なので，黄色で示した領域がそれぞれの格納場所となる.具体的には,int_array[0]=1 のアドレスが 0x7ffee97b4a30(領域は 0x7ffee97b4a30～0x7ffee97b4a33), int_array[1]=2 のアドレスが 0x7ffee97b4a34(領域は 0x7ffee97b4a34～0x7ffee97b4a37), int_array[2]=3 のアドレスが 0x7ffee97b4a38(領域は 0x7ffee97b4a38～0x7ffee97b4a3b)となる.

sizeof()により char のサイズは 1byte であることを確認した．文字型変数は cha=65;と設定し, address は, 0x7ffeec408a2f であり，水色で示した場所が格納場所となる.

文字型配列は char_array="ABC";と初期化を行い，整数型配列と同じように for 文で各値とアドレスを取り出した．文字列はそれぞれ 1byte なので，紫色で示したように値が格納されている．具体的には, char_array[0] = “A”のアドレスが 0x7ffee97b4a24, char_array[1] = “B”のアドレスが 0x7ffee97b4a25, char_array[2] = “C”のアドレスが 0x7ffee97b4a26 となっている．文字列配列の場合,¥0 が最後に置かれる.

最後にポインタ変数について説明する. sizeof()により pointer のサイズは 8byte であることを確認した．まず変数 a=5;を設定し,ポインタ変数 p に a=5 のアドレスを代入した. a のアドレスとポインタ変数 p の値のアドレスは 0x7ffee97b4a20 であり，図表では赤色で示した領域(0x7ffee97b4a20～0x7ffee97b4a23)が格納場所となる．また，ポインタ変数 p の指し示す場所に格納されている値は 5 となる．ポインタ変数 p のアドレスは 0x7ffee97b4a18 であり,であり,二重線で囲んだ領域が(0x7ffee97b4a18～0x7ffee97b4a1F)が格納場所となる.


0x7ffee97b4a4c
0x7ffee97b4a48
10
short_int = 10;
0x7ffee97b4a44
0x7ffee97b4a40
15
long_ing = 15;
0x7ffee97b4a3c
0x7ffee97b4a38
3
int_array[2] = {3};
0x7ffee97b4a34
2
int_array[1] = {2};
0x7ffee97b4a30
1
int_array[0] = {1};
0x7ffee97b4a2c
65
cha=65;
0x7ffee97b4a28
0x7ffee97b4a24
A
B
C
\0
char_array = "ABC";
0x7ffee97b4a20
5
a = 5; *p=5;
0x7ffee97b4a1c
0x7ffee97b4a18
&p=0x7ffeec408a18
0x7ffee97b4a14
0x7ffee97b4a10

Program

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

int main() {
   //Variable of integer type
   short short_int = 10;
   long long_int = 15;

   printf("int = %lu¥n", sizeof(int));

   printf("short_int = %d, &short_int = %p, sizeof_short_int = %lu¥n", short_int, &short_int, sizeof(short_int));
   printf("long_int = %ld, &long_int = %p, sizeof_long_ing = %lu¥n", long_int, &long_int, sizeof(long_int));

   //Array variables of integer type
   int i, int_array[3] = {1,2,3};
   for(i=0; i<3; i++){
      printf("int_array[%d] = %d, &int_array[%d] = %p¥n", i, int_array[i], i, &int_array[i]);
   }

   //Char variable
   char cha = 'A';
   printf("cha = %d, &cha = %p, sizeof_cha = %lu¥n", cha, &cha, sizeof(cha));

   //Character array
   int j;
   char char_array[4] = "ABC";
   for(j=0; j<3; j++){
      printf("char_array[%d] = %c, &char_array[%d] = %p¥n", j, char_array[j], j, &char_array[j]);
```

```
    }
    printf("char_array = %s¥n", char_array);

    //Assigning addresses to Pointer
    int a = 5;
    int *p;
    p = &a;
    printf("a = %d, &a = %p¥n", a, &a);
    printf("p = %p, *p = %d, sizeof_p = %lu¥n", p, *p, sizeof(p));
    printf("&p = %p¥n", &p);

    exit(0);

}
```

Output

int = 4
short_int = 10, &short_int = 0x7ffee97b4a4a, sizeof_short_int = 2
long_int = 15, &long_int = 0x7ffee97b4a40, sizeof_long_ing = 8
int_array[0] = 1, &int_array[0] = 0x7ffee97b4a30
int_array[1] = 2, &int_array[1] = 0x7ffee97b4a34
int_array[2] = 3, &int_array[2] = 0x7ffee97b4a38
cha = 65, &cha = 0x7ffee97b4a2f, sizeof_cha = 1
char_array[0] = A, &char_array[0] = 0x7ffee97b4a24
char_array[1] = B, &char_array[1] = 0x7ffee97b4a25
char_array[2] = C, &char_array[2] = 0x7ffee97b4a26
char_array = ABC
a = 5, &a = 0x7ffee97b4a20
p = 0x7ffee97b4a20, *p = 5, sizeof_p = 8
&p = 0x7ffee97b4a18