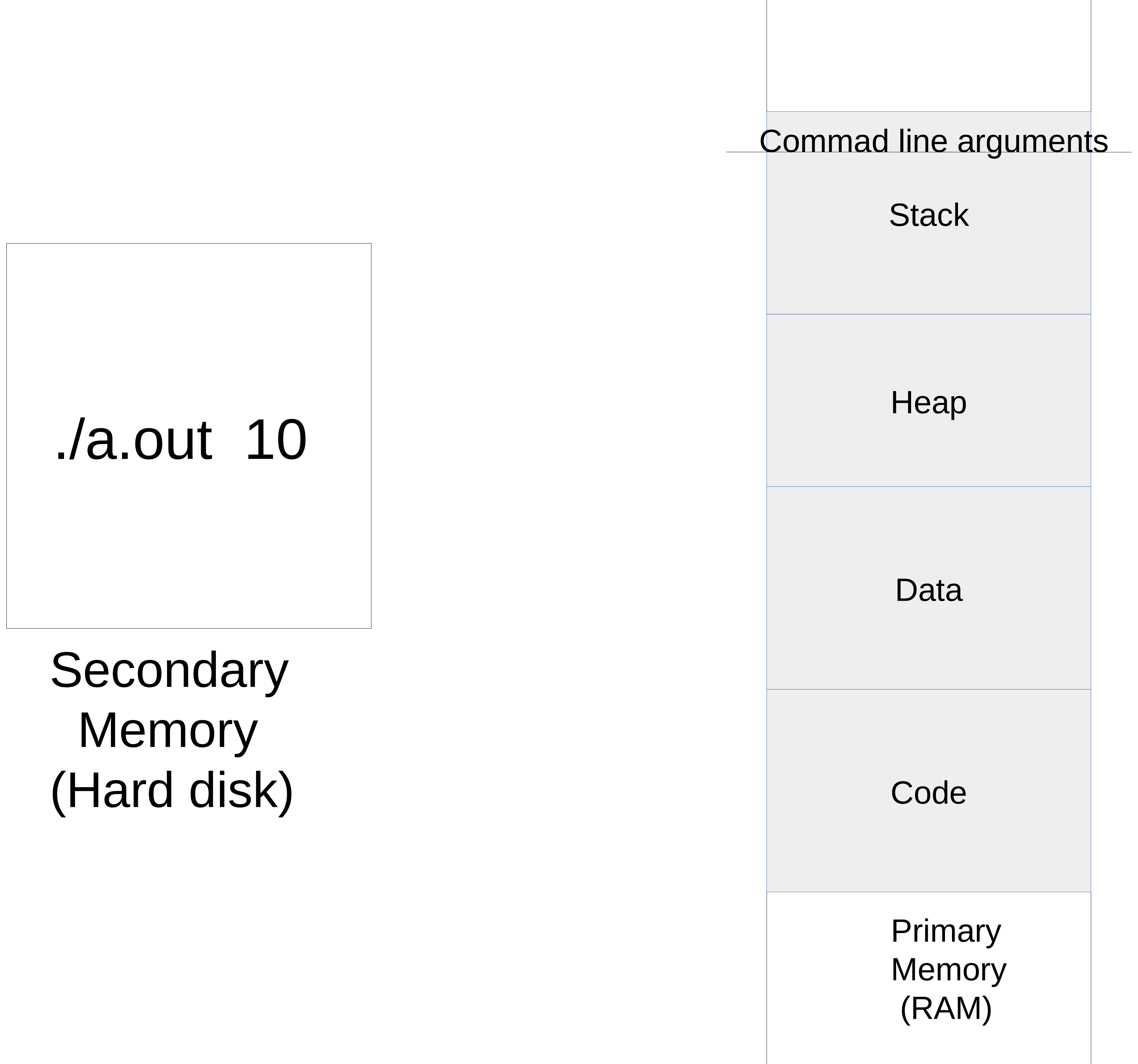
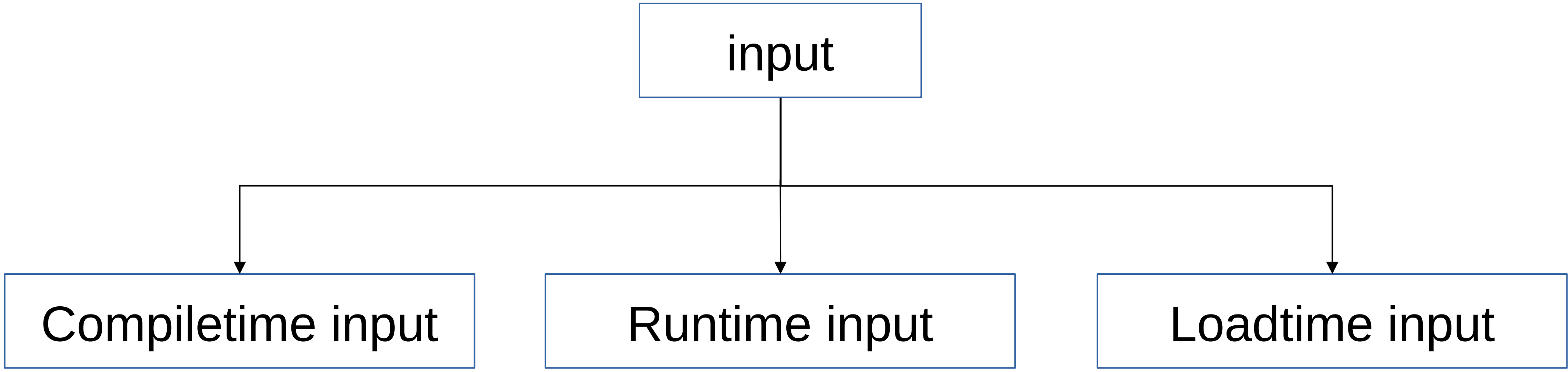

./a.out 10
Secondary Memory (Hard disk)
Commad line arguments
Stack
Heap
Data
Code
Primary Memory (RAM)



Ex : int x = 10;

Ex : scanf(“%d”,&x);

Ex : $./a.out 10;

Merit:
Faster execution.
Demerit :
Re-compilation is required
to modify the value.

Merit:
Re-compilation is not
Required to modify the value.
Demerit :
Slower in execution.

Merit:
Faster in execution & re-compilation is not
Required to modify the value.
Demerit :
Convertion functions are required to take
a proper input for integers and floats.

Command line arguments (loadtime input) :

Providing the arguments to a program at command prompt along with executable file is called as command line arguments.

These inputs we are giving to a program at the time of executable file loading into main memory. So we also call it as loadtime input.

Ex : ./a.out 1234   22.7   A   “vector” --> these arguments are received by main().

A main() can be written in 3 ways.

1. main()
2. main(int argc,char *argv[ ]); //for command line arguments
3. main(int argc,char *argv[ ], char *env[ ]); //for command line arg & environmental arguments

2. main(int argc,char *argv[ ]);

(or)

main(int argc,char **argv);

argc ---> argument count;
(no.of arugments at command prompt to a program.)

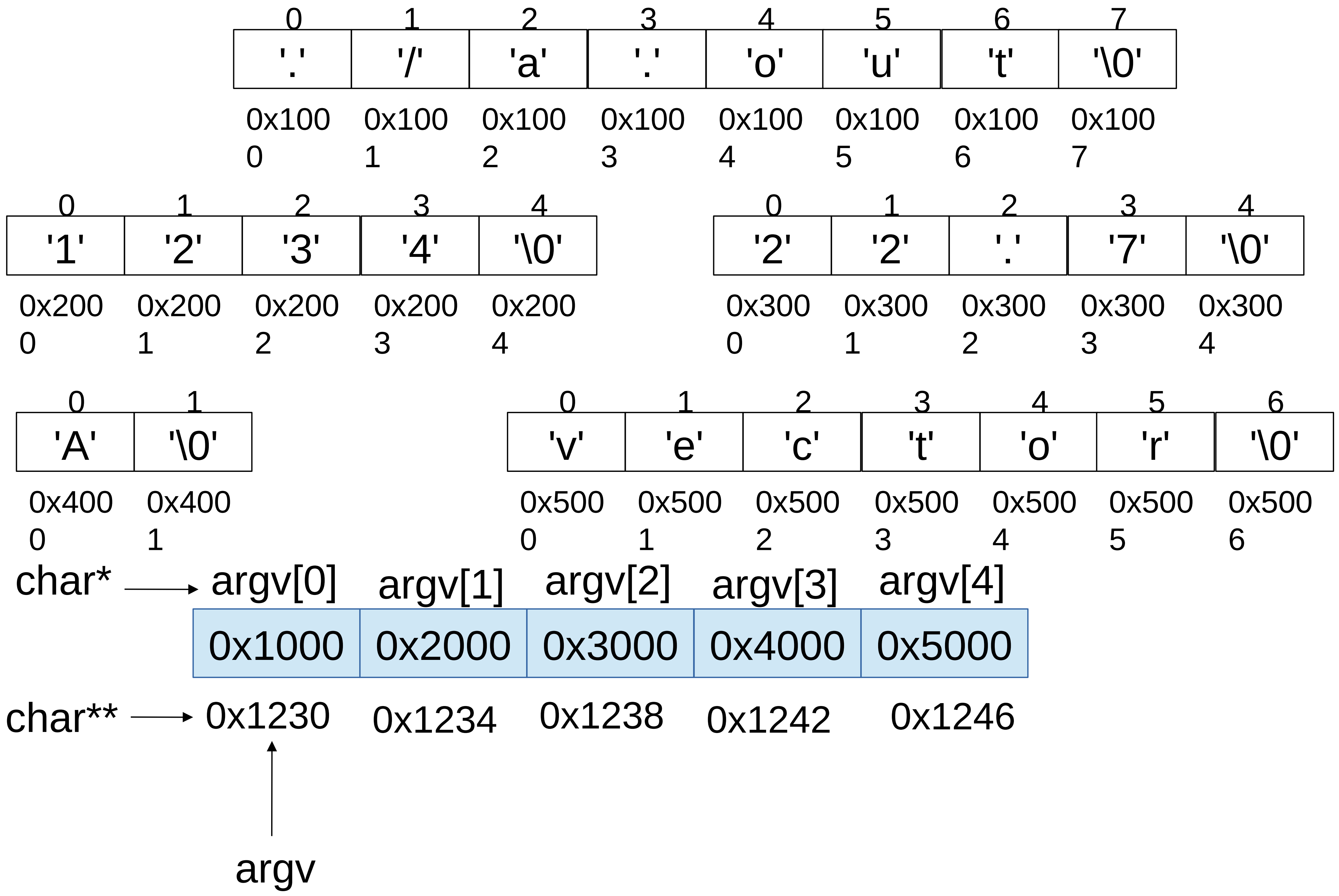
argv ---> argument vector; (all arugments are here)

Note : Bydefault command line arguments are treated as strings.

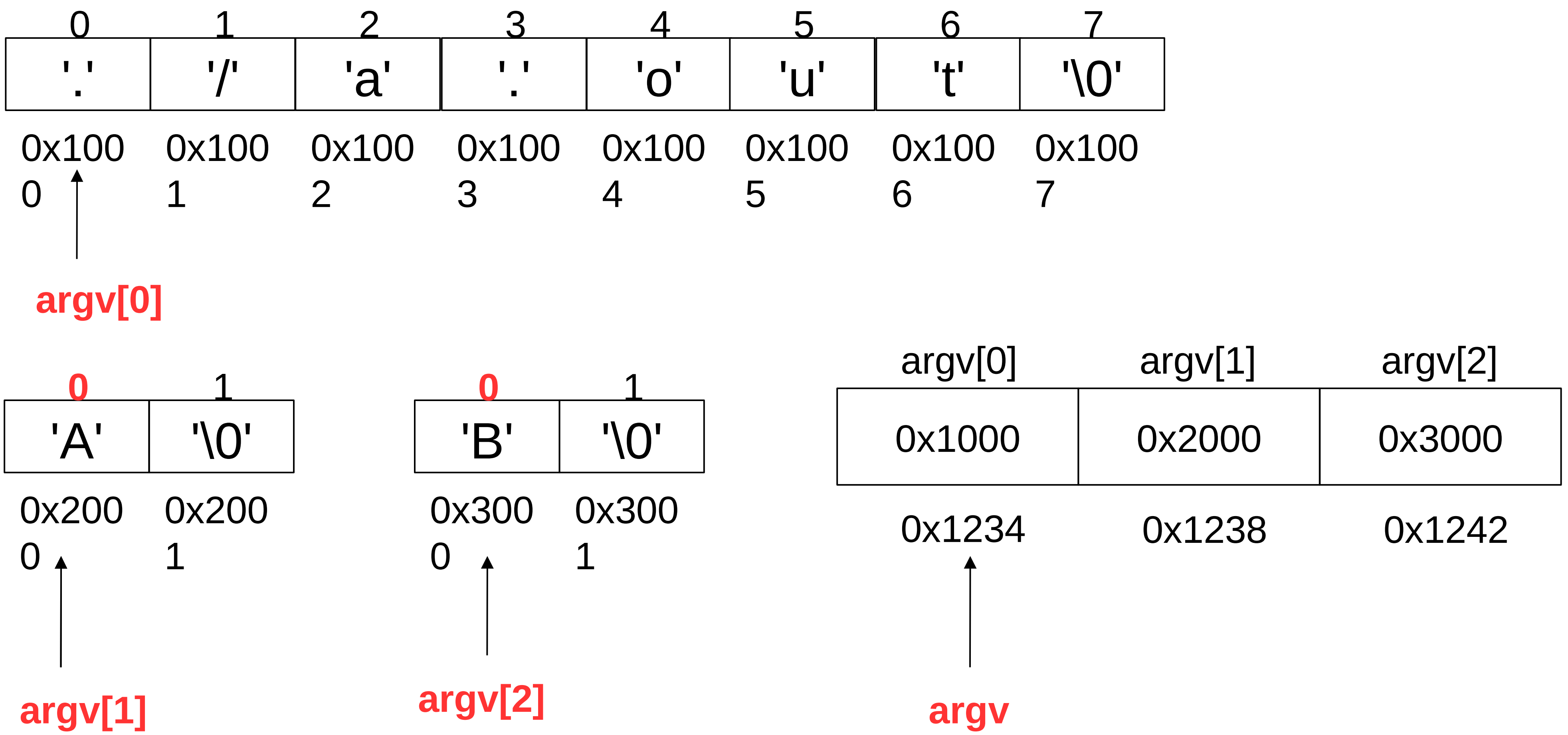
Ex : ./a.out 1234 22.7 A vector;

1.Here all the above arguments are treated as strings.
2.Bydefault a string itself represent as base address.
3.syntatical representation of string is **const char***
4. argv is pointer (char **) , which points a char array of pointer base addr.

```
./a.out  1234  22.7   A   vector
```


0 1 2 3 4 5 6 7
'.' '/' 'a' '.' 'o' 'u' 't' '\0'
0x1000 0x1001 0x1002 0x1003 0x1004 0x1005 0x1006 0x1007
0 1 2 3 4
'1' '2' '3' '4' '\0'
0x2000 0x2001 0x2002 0x2003 0x2004
0 1 2 3 4
'2' '2' '.' '7' '\0'
0x3000 0x3001 0x3002 0x3003 0x3004
0 1
'A' '\0'
0x4000 0x4001
0 1 2 3 4 5 6
'v' 'e' 'c' 't' 'o' 'r' '\0'
0x5000 0x5001 0x5002 0x5003 0x5004 0x5005 0x5006
char*
argv[0] argv[1] argv[2] argv[3] argv[4]
0x1000 0x2000 0x3000 0x4000 0x5000
char**
0x1230 0x1234 0x1238 0x1242 0x1246
argv

```
#include<stdio.h>
int main(int argc,char *argv[])
{
        printf("argc = %d\n",argc);
        int i;
        for(i=0;i<argc;i++)
        printf("argv[%d] = %s\n",i,argv[i]);
}
```

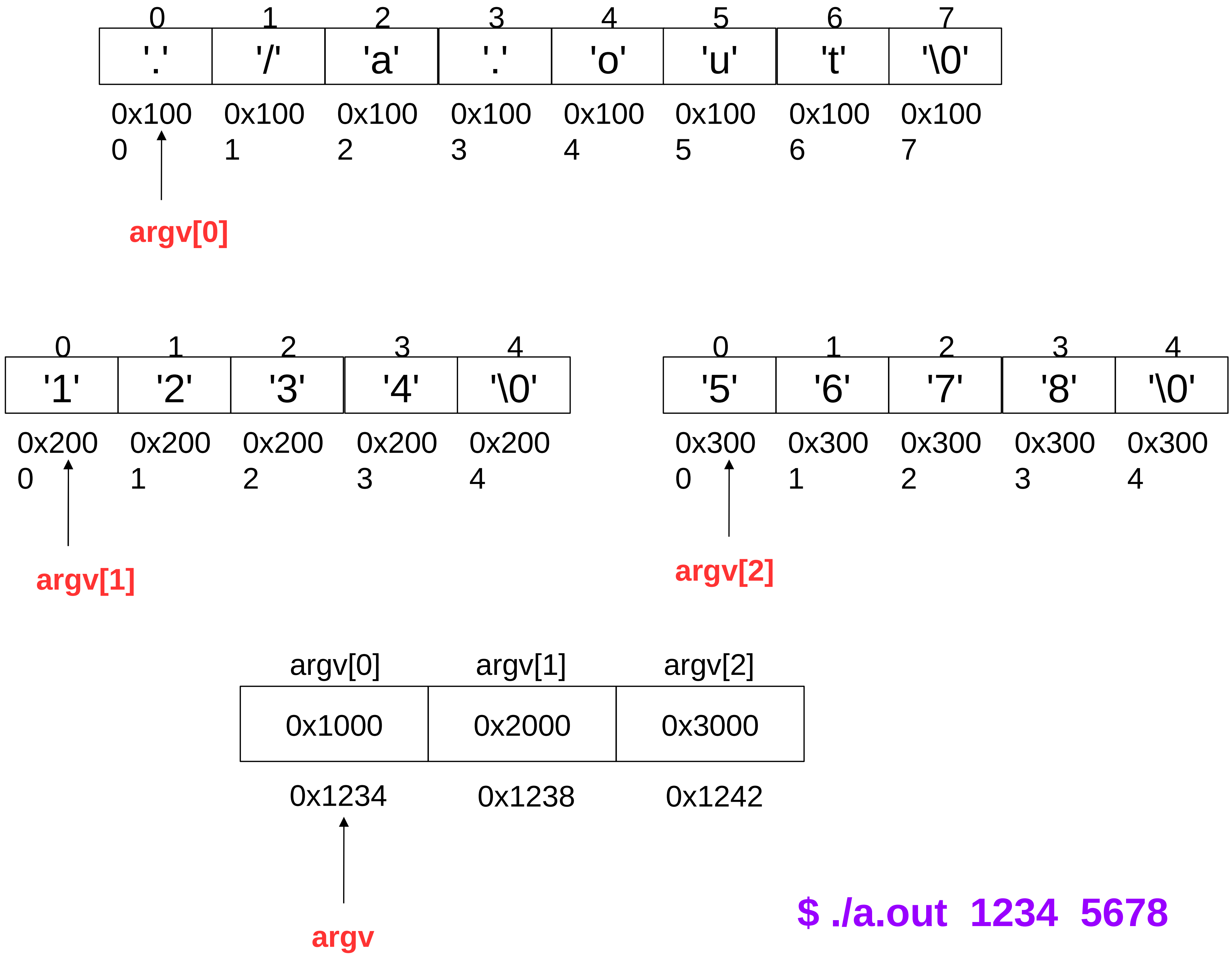


```
$ ./a.out  A   B
```

'A' ------> argv[1][0]   or   *argv[1]

'B' ------> argv[2][0]   or   *argv[2]

```c
//write a program to provide a char input to a program at commad prompt.
#include<stdio.h>
int main(int argc,char *argv[])
{
        if(argc != 3) {
        printf("Usage : ./a.out char char\n");
        return 0;
        }

         char ch1,ch2;
         ch1 = argv[1][0];
         ch2 = argv[2][0];

         printf("ch1 = %c  ch2 = %c\n",ch1,ch2);
}
// $ ./a.out A B
```



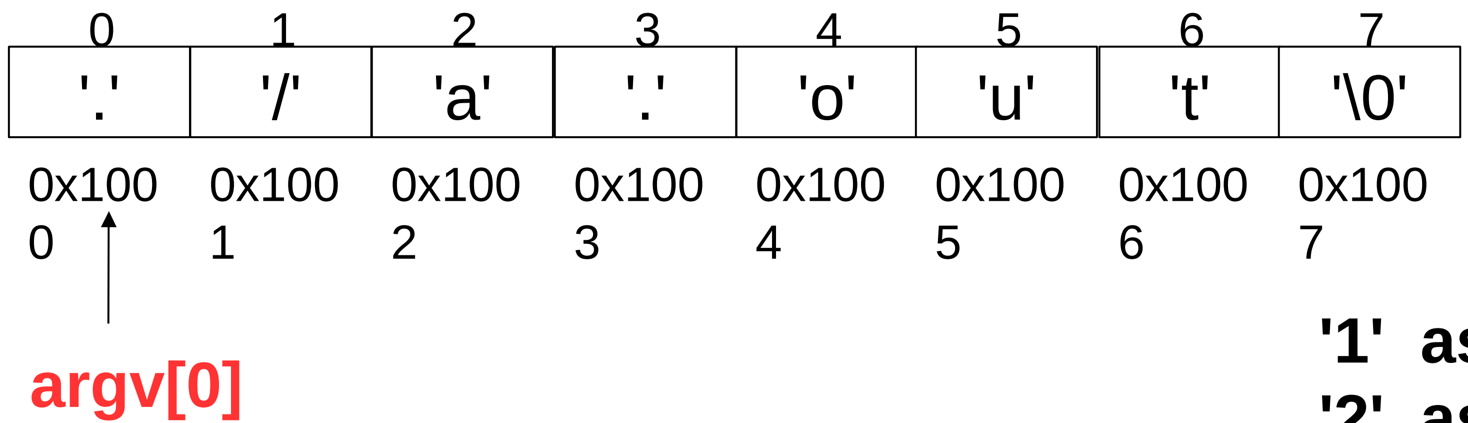
```
$ ./a.out  1234  5678
```

```
//write a program to provide integer input to a program
#include<stdio.h>
int main(int argc,char **argv)
{
        if(argc != 3) {
        printf("Usage : ./a.out int int\n");
        return 0;
        }

        int x,y;
        //x = argv[1];    // x = string
        x = atoi(argv[1]); // x = int
        y = atoi(argv[2]);

        printf("x = %d  y = %d\n",x,y);
}
//$ ./a.out 1234 4567
```

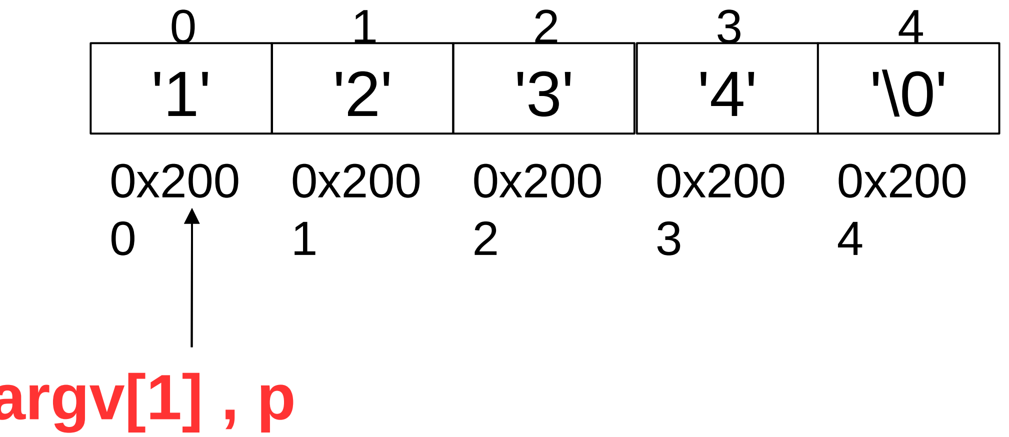
```
//write a program to provide float input to a program
#include<stdio.h>
#include<stdlib.h>
int main(int argc,char **argv)
{
        if(argc != 3) {
        printf("Usage : ./a.out float float\n");
        return 0;
        }

        float x,y;
        x = atof(argv[1]);
        y = atof(argv[2]);

        printf("x = %f  y = %f\n",x,y);
}
//$ ./a.out 1234 4567
```

//write a program to provide all types of inputs to a program using command line arguments.

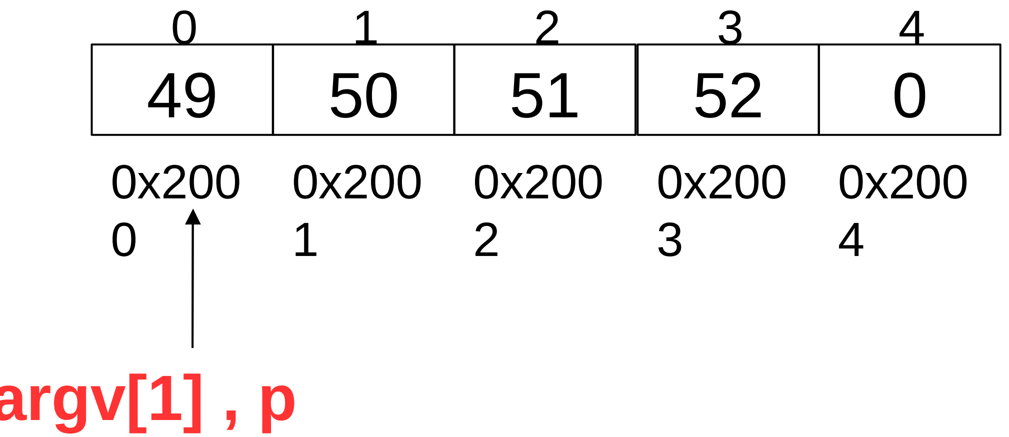
```c
#include<stdio.h>
#include<stdlib.h>
#include<string.h>
int main(int argc,char *argv[])
{
  if(argc != 5) {
  printf("Usage : ./a.out char int float string\n");
  return 0;
  }
  char ch; int x; float f; char s[20];
  ch = argv[1][0];
  x = atoi(argv[2]);
  f = atof(argv[3]);
  //s = argv[4];  //base addr = base addr
  strcpy(s,argv[4]);

  printf("ch = %c\n",ch);
  printf("x = %d\n",x);
  printf("f = %f\n",f);
  printf("s = %s\n",s);
}
```

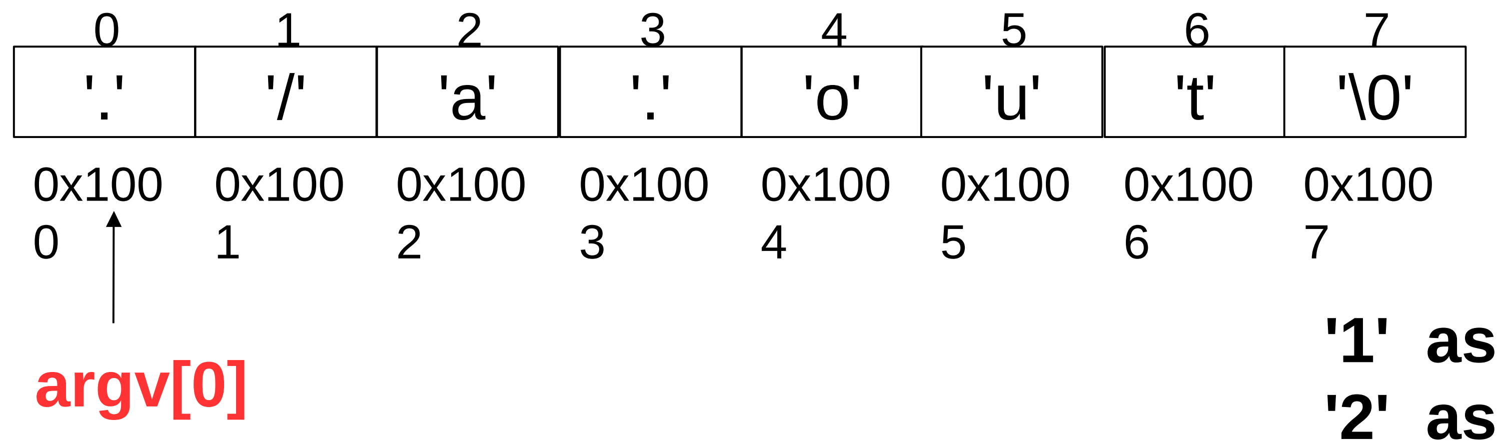

0 1 2 3 4 5 6 7
'.' '/' 'a' '.' 'o' 'u' 't' '\0'
0x1000 0x1001 0x1002 0x1003 0x1004 0x1005 0x1006 0x1007
argv[0]

'1' ascii value ---> 49
'2' ascii value ---> 50
'3' ascii value ---> 51
'4' ascii value ---> 52
'\0' ascii value ---> 0

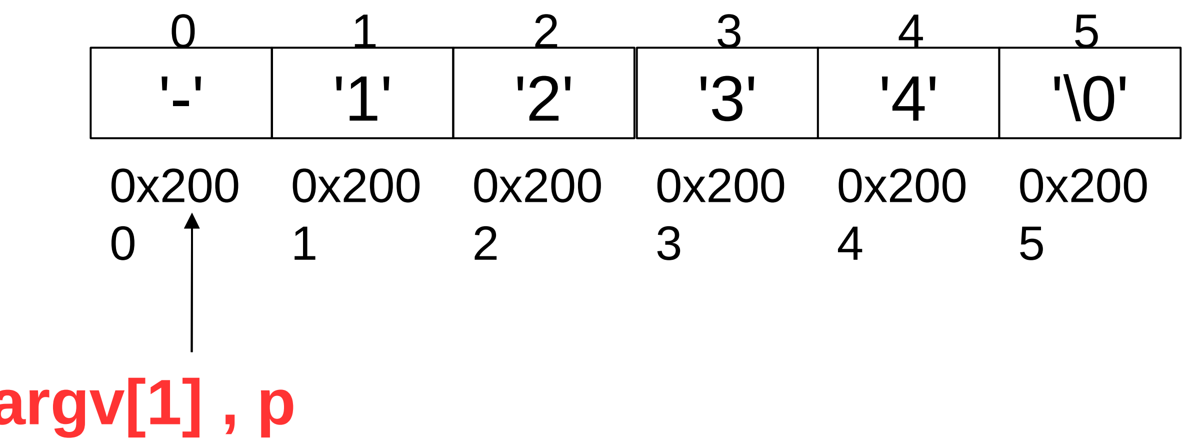


```
$ ./a.out 1234
```

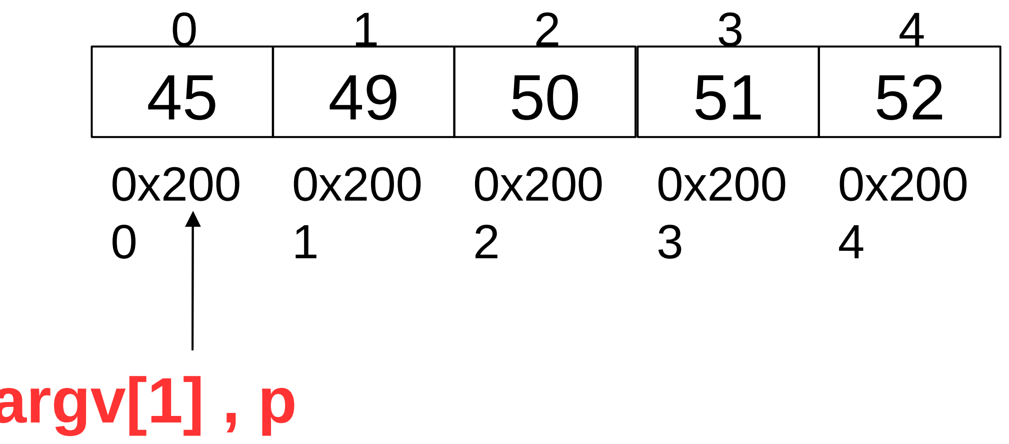


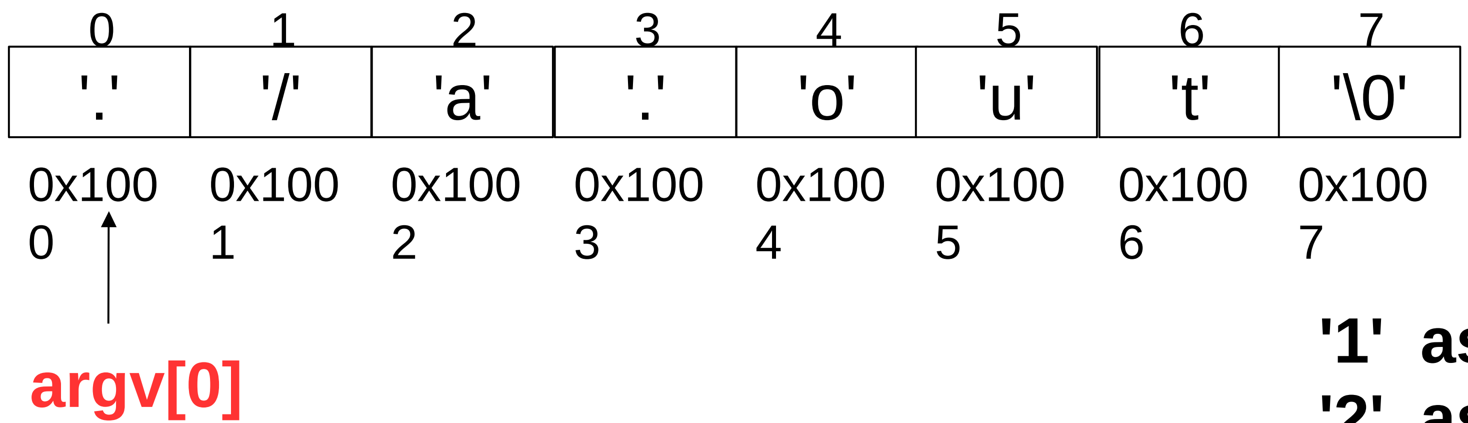

0 1 2 3 4 5 6 7
'.' '/' 'a' '.' 'o' 'u' 't' '\0'
0x1000 0x1001 0x1002 0x1003 0x1004 0x1005 0x1006 0x1007
argv[0]

'1' ascii value ---> 49
'2' ascii value ---> 50
'3' ascii value ---> 51
'4' ascii value ---> 52
'\0' ascii value ---> 0

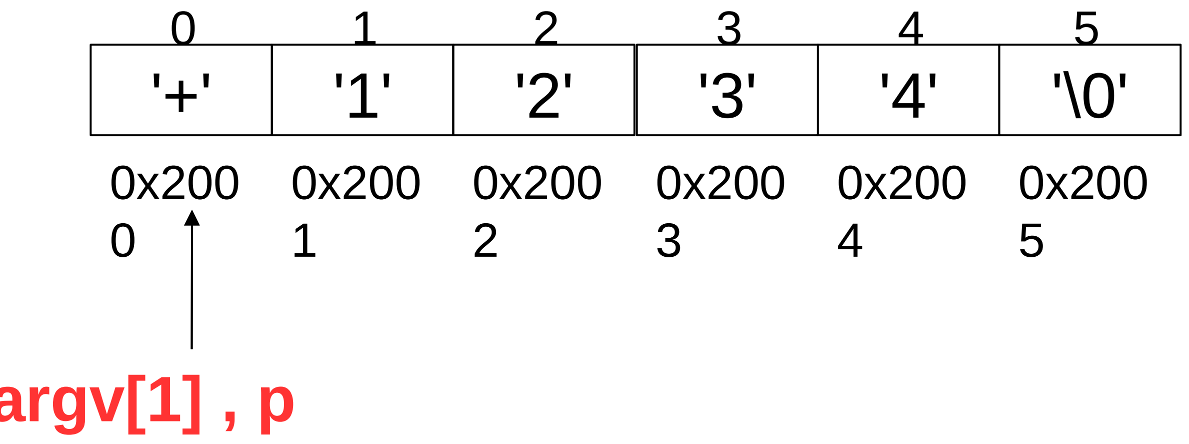


```
$ ./a.out -1234
```

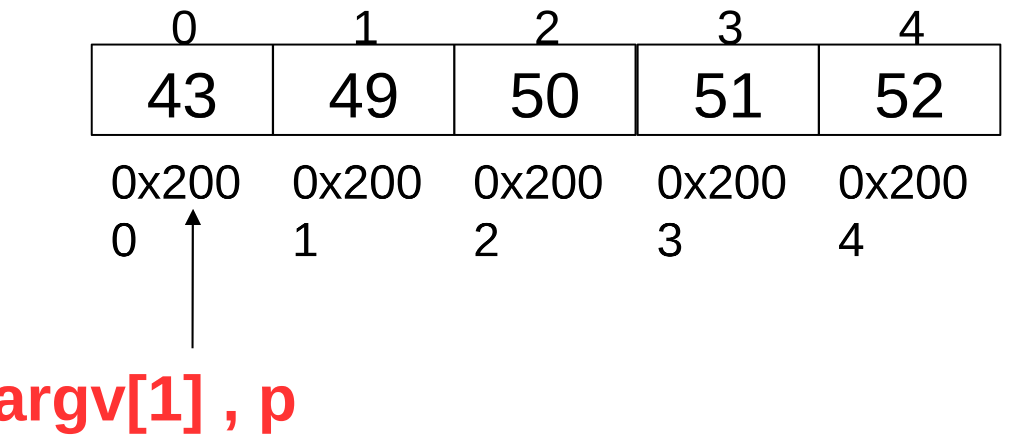



0 1 2 3 4 5 6 7
'.' '/' 'a' '.' 'o' 'u' 't' '\0'
0x1000 0x1001 0x1002 0x1003 0x1004 0x1005 0x1006 0x1007
argv[0]

'1' ascii value ---> 49
'2' ascii value ---> 50
'3' ascii value ---> 51
'4' ascii value ---> 52
'\0' ascii value ---> 0



```
$ ./a.out +1234
```



```
P[0] – 48 = 49 – 48
          = 1;

P[1] – 48 = 49 – 48
          = 2;

P[2] – 48 = 49 – 48
          = 3;

P[3] – 48 = 49 – 48
          = 4;
```

atoi() logic :

```
int sum = 0;
for(i=0;p[i];i++)
{
    sum = sum * 10 + p[i] – 48;
}

return sum;
```

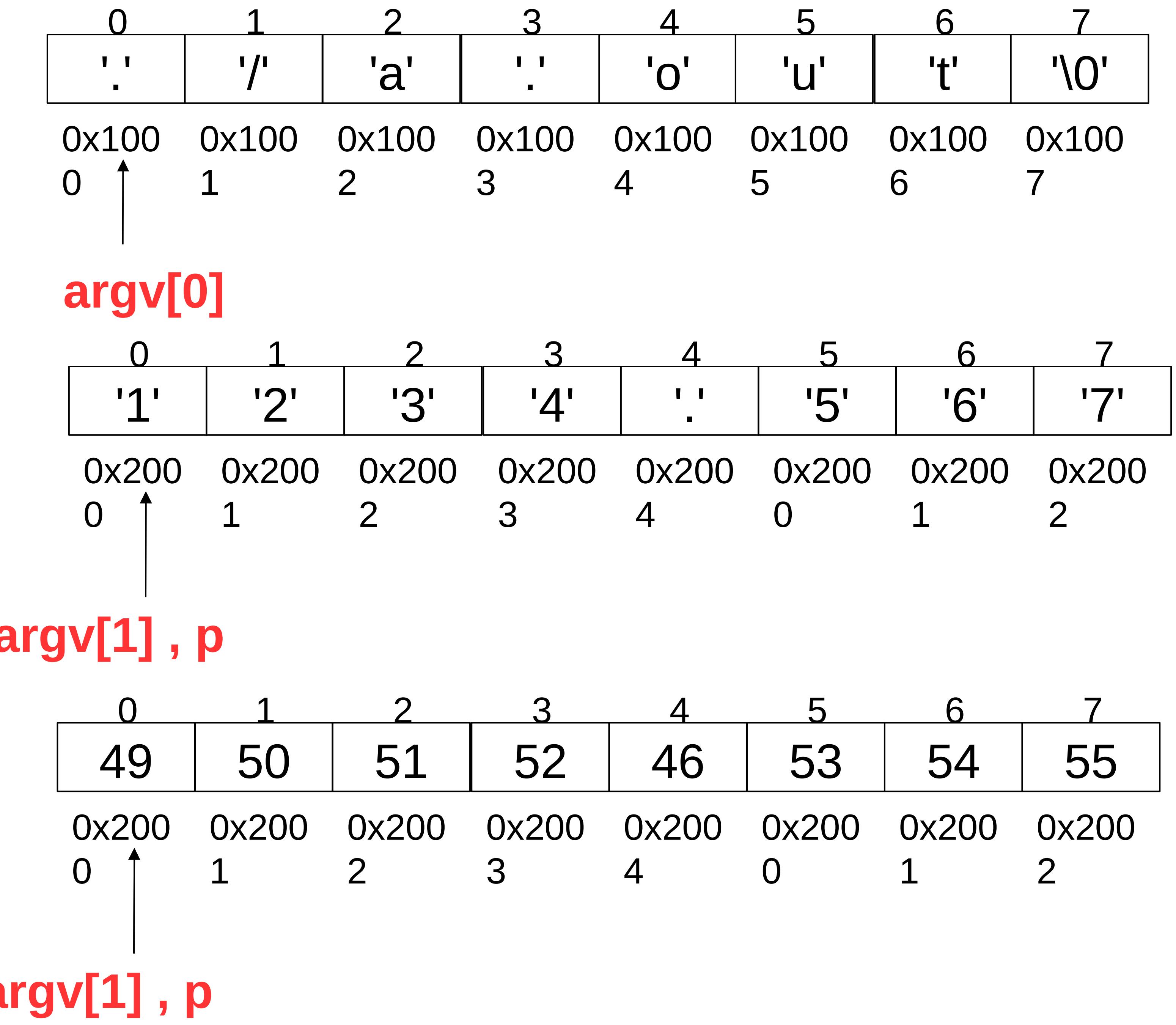
```
sum * 10+(p[i]-48) = sum;

0   * 10+1  =  1;
1   * 10+2  =  12;
12  * 10+3  = 123;
123 * 10+4  = 1234;
```

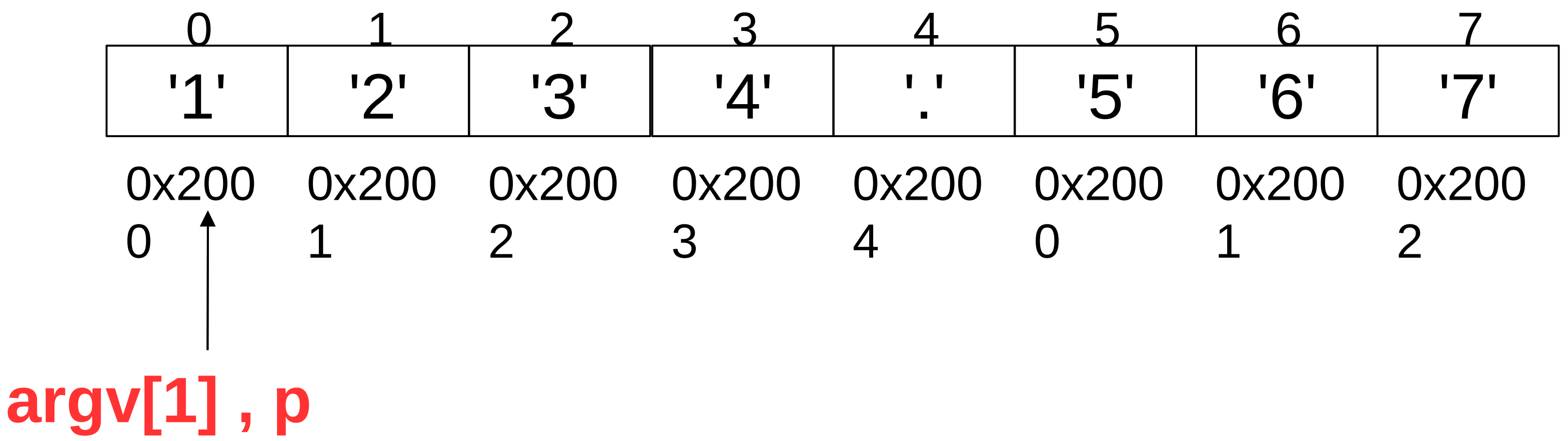
```
//write a program to provide integer input to a program
#include<stdio.h>
int my_atoi(const char *p);
int main(int argc,char **argv)
{
        int x;
        if(argc != 2) {
        printf("Usage : ./a.out int\n");
        return 0;
         }
         x = my_atoi(argv[1]);
         printf("x = %d\n",x);
}
```

```c
int my_atoi(const char *p)
{
        int i = 0,sum = 0;
        if((p[0] == '-')||(p[0] == '+'))
        i = 1;

        for(;p[i];i++)
        {
                if((p[i]>='0')&&(p[i]<='9'))
                sum = sum*10+(p[i]-48);
                else
                break;
        }
        if(p[0] == '-')
        return -sum;
        else
        return sum;
}
```

atof()

```
$ ./a.out 1234.567
```


0 1 2 3 4 5 6 7
'.' '/' 'a' '.' 'o' 'u' 't' '\0'
0x1000 0x1001 0x1002 0x1003 0x1004 0x1005 0x1006 0x1007
argv[0]
0 1 2 3 4 5 6 7
'1' '2' '3' '4' '.' '5' '6' '7'
0x2000 0x2001 0x2002 0x2003 0x2004 0x2000 0x2001 0x2002
argv[1] , p
0 1 2 3 4 5 6 7
49 50 51 52 46 53 54 55
0x2000 0x2001 0x2002 0x2003 0x2004 0x2000 0x2001 0x2002
argv[1] , p



‘1’ ---> 49 – 48 ----> 1
‘2’ ---> 50 – 48 ----> 2
‘3’ ---> 51 – 48 ----> 3
‘4’ ---> 52 – 48 ----> 4

‘5’ ----> 53 – 48 ---> 5 ---> 5 * 0.1 ----> 0.5
‘6’ ----> 54 – 48 ---> 6 ---> 6 * 0.01 ---> 0.06
‘7’ ----> 55 – 48 ----> 7 --> 7 * 0.001 --> 0.007

0.567

```
$ ./a.out 1234.567
```

```
0     * 10+1  =  1;
1     * 10+2  =  12;
12   * 10+3  =  123;
123 * 10+4  =  1234;


5 --> 0.5
6 --> 0.06
7 --> 0.007  (+)
        ---------
         0.567
        ---------

         0 +  0.1  *  5    =  0.5;
      0.5 + 0.01  *  6   =  0.56;
    0.56 + 0.001 * 7   =  0.567;

    sum2 + f * (p[i]-48) = sum2;      ,  f = f * 0.1;

         0.1 * 0.1 = 0.01,    0.01 * 0.1 = 0.001
```

```
//write a program to provide integer input to a program
#include<stdio.h>
float my_atof(const char *p);
int main(int argc,char **argv)
{
        float x;
        if(argc != 2) {
        printf("Usage : ./a.out int\n");
        return 0;
         }
         x = my_atof(argv[1]);
         printf("x = %f\n",x);
}
```

```
float my_atof(const char *p)
{
        int i = 0,sum1 = 0;
        float sum2 = 0, f = 0.1;
        if((p[0] == '-')||(p[0] == '+'))
        i = 1;

        for(;p[i] != '.';i++)
        {
                if((p[i]>='0')&&(p[i]<='9'))
                sum1 = sum1*10+(p[i]-48);
                else
                break;
        }

        for(i = i+1;p[i];i++,f = f*0.1)
        {
                if((p[i]>='0')&&(p[i]<='9'))
                sum2 = sum2+f*(p[i]-48);
        }

        if(p[0] == '-')
        return -(sum1+sum2);
        else
        return sum1+sum2;
```

```c
//write a program to implement basic calculator program using command line arguments
#include<stdio.h>
int main(int argc,char *argv[])
{
        if(argc != 4) {
        printf("Usage : ./a.out int int op\n");
        return 0;
        }

         int x,y,z;
         x = atoi(argv[1]);
         y = atoi(argv[2]);

         switch(argv[3][0])
         {
                 case '+' : z = x+y; break;
                 case '-' : z = x-y; break;
                 case '*' : z = x*y; break;
                 case '/' : z = x/y; break;
                 case '%' : z = x%y; break;
                 default : printf("Invalid option...\n");
                           return 0;
         }
                 printf("z = %d\n",z);
}
```

```
$ ./cal 10 20  '*'
```