

Problem Set #5

1. Write a program that prints the program arguments. For example, if the program named **prog** is launched by the command:

```
> prog -a 1 -b 2 -c 3 4 -d "7 "
```

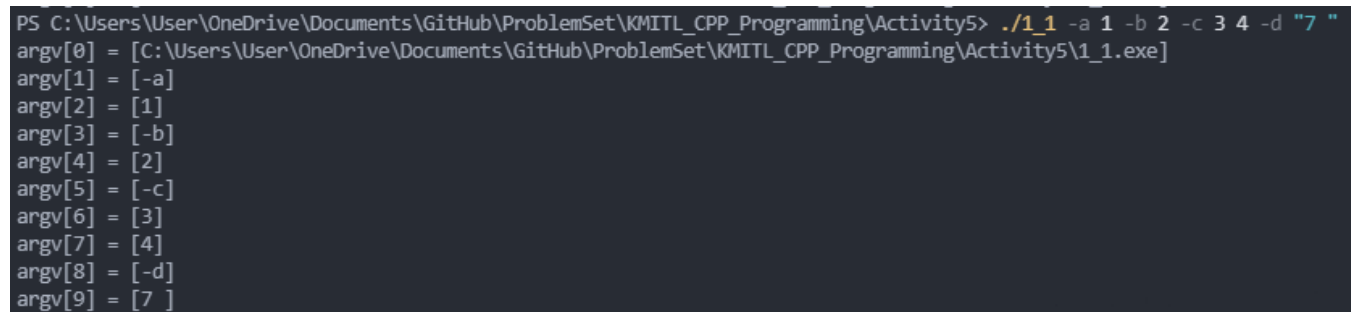
The output will be as follows.

```
argv[0] = [prog]
argv[1] = [-a]
argv[2] = [1]
argv[3] = [-b]
argv[4] = [2]
argv[5] = [-c]
argv[6] = [3]
argv[7] = [4]
argv[8] = [-d]
argv[9] = [7 ]
```

- 1.1) Write the program using an array subscript operator to access each argument value.

```
#include <stdio.h>

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



The screenshot shows a terminal window with the following content:

```
PS C:\Users\User\OneDrive\Documents\GitHub\ProblemSet\KMITL_CPP_Programming\Activity5> ./1_1 -a 1 -b 2 -c 3 4 -d "7 "
argv[0] = [C:\Users\User\OneDrive\Documents\GitHub\ProblemSet\KMITL_CPP_Programming\Activity5\1_1.exe]
argv[1] = [-a]
argv[2] = [1]
argv[3] = [-b]
argv[4] = [2]
argv[5] = [-c]
argv[6] = [3]
argv[7] = [4]
argv[8] = [-d]
argv[9] = [7 ]
```

Name: Dulapah Vibulsanti
ID: 64011388

1.2) Write the program using a pointer to access each argument value.

```
#include <stdio.h>

int main(int argc, char** argv) {
    for (char** i = argv; i < argv + argc; i++)
        printf("argv[%d] = [%s]\n", i - argv, *i);
    return 0;
}
```

```
PS C:\Users\User\OneDrive\Documents\GitHub\ProblemSet\KMITL_CPP_Programming\Activity5> ./1_2 -a 1 -b 2 -c 3 4 -d "7 "
argv[0] = [C:\Users\User\OneDrive\Documents\GitHub\ProblemSet\KMITL_CPP_Programming\Activity5\1_2.exe]
argv[1] = [-a]
argv[2] = [1]
argv[3] = [-b]
argv[4] = [2]
argv[5] = [-c]
argv[6] = [3]
argv[7] = [4]
argv[8] = [-d]
argv[9] = [7 ]
```

1.3) Change the program so that it prints the arguments in reverse from item #9 to item #1 excluding item #0. Use an array subscript operator to access each argument value.

```
#include <stdio.h>

int main(int argc, char** argv) {
    for (int i = argc - 1; i >= 0; i--)
        printf("argv[%d] = [%s]\n", i, argv[i]);
    return 0;
}
```

```
PS C:\Users\User\OneDrive\Documents\GitHub\ProblemSet\KMITL_CPP_Programming\Activity5> ./1_3 -a 1 -b 2 -c 3 4 -d "7 "
argv[9] = [7 ]
argv[8] = [-d]
argv[7] = [4]
argv[6] = [3]
argv[5] = [-c]
argv[4] = [2]
argv[3] = [-b]
argv[2] = [1]
argv[1] = [-a]
argv[0] = [C:\Users\User\OneDrive\Documents\GitHub\ProblemSet\KMITL_CPP_Programming\Activity5\1_3.exe]
```

Name: Dulapah Vibulsanti
ID: 64011388

1.4) Change the program from 1.3) to use a pointer to access each argument value.

```
#include <stdio.h>

int main(int argc, char** argv) {
    for (char** i = argv + argc - 1; i >= argv; i--)
        printf("argv[%d] = [%s]\n", i - argv, *i);
    return 0;
}
```

```
PS C:\Users\User\OneDrive\Documents\GitHub\ProblemSet\KMITL_CPP_Programming\Activity5> ./1_4 -a 1 -b 2 -c 3 4 -d "7 "
argv[9] = [7 ]
argv[8] = [-d]
argv[7] = [4]
argv[6] = [3]
argv[5] = [-c]
argv[4] = [2]
argv[3] = [-b]
argv[2] = [1]
argv[1] = [-a]
argv[0] = [C:\Users\User\OneDrive\Documents\GitHub\ProblemSet\KMITL_CPP_Programming\Activity5\1_4.exe]
```

Name: Dulapah Vibulsanti
ID: 64011388

2. Write a program that dump the contents of a file in binary mode. Each line should not print more than 8 bytes of data.
- 2.1) Dump each byte value in hexadecimal.

input	Output
Hello	48 65 6C 6C 6F 0D 0A 57
World	6F 72 6C 64 0D 0A

```
#include <stdio.h>

int main() {
    FILE* in_file = fopen("2_1in.txt", "rb");
    int count = 0;
    for (int c; (c = fgetc(in_file)) != EOF;) {
        if (count >= 8) {
            printf("\n");
            count = 0;
        }
        printf("%02X ", c);
        count++;
    }
    fclose(in_file);
    return 0;
}
```

```
48 65 6C 6C 6F 0D 0A 57
6F 72 6C 64 0D 0A
```

Name: Dulapah Vibulsanti
ID: 64011388

2.2) Dump each byte value in octal.

input	Output
Hello	0110 0145 0154 0154 0157 0015 0012 0127
World	0157 0162 0154 0144 0015 0012

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

```
int main() {  
    FILE* in_file = fopen("2_2in.txt", "rb");  
    int count = 0;  
    for (int c; (c = fgetc(in_file)) != EOF;) {  
        if (count >= 8) {  
            printf("\n");  
            count = 0;  
        }  
        printf("%04o ", c);  
        count++;  
    }  
    fclose(in_file);  
    return 0;  
}
```

```
0110 0145 0154 0154 0157 0015 0012 0127  
0157 0162 0154 0144 0015 0012
```

Name: Dulapah Vibulsanti
ID: 64011388

2.3) Dump each byte value in decimal.

input	Output
Hello	72 101 108 108 111 13 10 87
World	111 114 108 100 13 10

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

```
int main() {  
    FILE* in_file = fopen("2_3in.txt", "rb");  
    int count = 0;  
    for (int c; (c = fgetc(in_file)) != EOF;) {  
        if (count >= 8) {  
            printf("\n");  
            count = 0;  
        }  
        printf("%d ", c);  
        count++;  
    }  
    fclose(in_file);  
    return 0;  
}
```

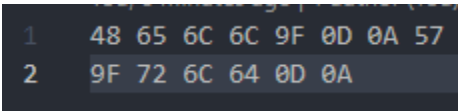
```
72 101 108 108 111 13 10 87  
111 114 108 100 13 10
```

2.4) Rewrite 2.1) by using only `fputc` to dump the data and not using `printf/fprintf`.

```
#include <stdio.h>

int main() {
    FILE* in_file = fopen("2_4in.txt", "rb");
    FILE* out_file = fopen("2_4out.txt", "wb");
    int firstDigit, secondDigit, thirdDigit;
    int count = 0;
    char hex[] = {'A', 'B', 'C', 'D', 'E', 'F'};
    for (int c; (c = fgetc(in_file)) != EOF;) {
        if (count >= 8) {
            fputc('\n', out_file);
            count = 0;
        }
        if (c >= 'A' && c <= 'Z') {
            if (c >= 'A' && c <= 'I') {
                c -= 24;
                firstDigit = (c / 10) + 48;
                secondDigit = (c % 10) + 48;
                fputc(firstDigit, out_file);
                fputc(secondDigit, out_file);
            }
            else if (c >= 'P' && c <= 'Y') {
                c -= 30;
                firstDigit = (c / 10) + 48;
                secondDigit = (c % 10) + 48;
                fputc(firstDigit, out_file);
                fputc(secondDigit, out_file);
            }
            else if ((c >= 'J' && c <= 'O') || c == 'Z') {
                c -= 102;
                firstDigit = c + 48;
                fputc(firstDigit, out_file);
                fputc(hex[((c + 102) % 16) - 10], out_file);
            }
            fputc(' ', out_file);
        }
        else if (c >= 'a' && c <= 'z') {
            if (c >= 'a' && c <= 'i') {
                c -= 36;
                firstDigit = (c / 10) + 48;
                secondDigit = (c % 10) + 48;
                fputc(firstDigit, out_file);
                fputc(secondDigit, out_file);
            }
            else if (c >= 'p' && c <= 'y') {
                c -= 42;
            }
        }
    }
}
```

```
        firstDigit = (c / 10) + 48;
        secondDigit = (c % 10) + 48;
        fputc(firstDigit, out_file);
        fputc(secondDigit, out_file);
    }
    else if ((c >= 'j' && c <= 'o') || c == 'z') {
        c -= 102;
        firstDigit = c + 48;
        fputc(firstDigit, out_file);
        fputc(hex[((c + 102) % 16) - 10], out_file);
    }
    fputc(' ', out_file);
}
else if (c == '\n') {
    int arr[6] = {'0', 'D', ' ', '0', 'A', ' '};
    for (int i = 0; i < 6; i++)
        fputc(arr[i], out_file);
}
count++;
}
fclose(in_file);
fclose(out_file);
return 0;
}
```



```
1  48 65 6C 6C 9F 0D 0A 57
2  9F 72 6C 64 0D 0A
```


2.5) Rewrite 2.2) by using only `fputc` to dump the data and not using `printf/fprintf`.

```
#include <stdio.h>

void separate_digit(int c, int* first, int* second, int* third) {
    *first = (c / 100);
    *second = ((c / 10) % 10);
    *third = (c % 10);
}

int main() {
    FILE* in_file = fopen("2_5in.txt", "rb");
    FILE* out_file = fopen("2_5out.txt", "wb");
    int firstDigit, secondDigit, thirdDigit;
    int count = 0;
    for (int c; (c = fgetc(in_file)) != EOF;) {
        if (count >= 8) {
            fputc('\n', out_file);
            count = 0;
        }
        if (c >= 'A' && c <= 'Z') {
            if (c >= 'A' && c <= 'G')
                c += 36;
            else if (c >= 'H' && c <= 'O')
                c += 38;
            else if (c >= 'P' && c <= 'W')
                c += 40;
            separate_digit(c, &firstDigit, &secondDigit, &thirdDigit);
            fputc('0', out_file);
            fputc(firstDigit + 48, out_file);
            fputc(secondDigit + 48, out_file);
            fputc(thirdDigit + 48, out_file);
            fputc(' ', out_file);
        }
        else if (c >= 'a' && c <= 'z') {
            if (c >= 'a' && c <= 'g')
                c += 44;
            else if (c >= 'h' && c <= 'o')
                c += 46;
            else if (c >= 'p' && c <= 'w')
                c += 48;
            separate_digit(c, &firstDigit, &secondDigit, &thirdDigit);
            fputc('0', out_file);
            fputc(firstDigit + 48, out_file);
            fputc(secondDigit + 48, out_file);
            fputc(thirdDigit + 48, out_file);
            fputc(' ', out_file);
        }
    }
}
```

Name: Dulapah Vibulsanti

ID: 64011388

```
        else if (c == '\n') {
            int arr[] = {'0', '0', '1', '5', ' ', '0', '0', '1', '2', ' '};
            for (int i = 0; i < 10; i++)
                fputc(arr[i], out_file);
        }
        count++;
    }
    fclose(in_file);
    fclose(out_file);
    return 0;
}
```

```
1  0110 0145 0154 0154 0157 0015 0012 0127
2  0157 0162 0154 0144 0015 0012
```

Name: Dulapah Vibulsanti
ID: 64011388

2.6) Rewrite 2.3) by using only `fputc` to dump the data and not using `printf/fprintf`.

```
#include <stdio.h>

int main() {
    FILE* in_file = fopen("2_6in.txt", "rb");
    FILE* out_file = fopen("2_6out.txt", "wb");
    int firstDigit, secondDigit, thirdDigit;
    int count = 0;
    for (int c; (c = fgetc(in_file)) != EOF;) {
        if (count >= 8) {
            fputc('\n', out_file);
            count = 0;
        }
        if (c >= 10 && c < 100) {
            firstDigit = (c / 10) + 48;
            secondDigit = (c % 10) + 48;
            fputc(firstDigit, out_file);
            fputc(secondDigit, out_file);
        }
        else if (c >= 100) {
            firstDigit = (c / 100) + 48;
            secondDigit = ((c / 10) % 10) + 48;
            thirdDigit = (c % 10) + 48;
            fputc(firstDigit, out_file);
            fputc(secondDigit, out_file);
            fputc(thirdDigit, out_file);
        }
        fputc(' ', out_file);
        count++;
    }
    fclose(in_file);
    fclose(out_file);
    return 0;
}
```

```
1  72 101 108 108 111 13 10 87
2  111 114 108 100 13 10 |
```

3. Suppose we have the following data:

ID	Name	Sub1	Sub2	Sub3
21000001	Joe	72.5	85.2	59.4
21000002	John	62.2	75.1	84.2
21000003	Alice	82.4	65.4	91.1
21000004	Bob	78.1	90.3	71.5
21000005	Mary	90.3	82.5	61.3

3.1) Define an array of the type that can store the above data and write a program to calculate average score for **Sub1**, **Sub2**, and **Sub3**.

Sub1 average = 77.1
Sub2 average = 79.7
Sub3 average = 73.5

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

struct Student {
    int id;
    char name[30];
    float sub1, sub2, sub3;
};

int main() {
    struct Student student[5];

    student[0].id = 21000001;
    strcpy(student[0].name, "Joe");
    student[0].sub1 = 72.5;
    student[0].sub2 = 85.2;
    student[0].sub3 = 59.4;

    student[1].id = 21000002;
    strcpy(student[1].name, "John");
    student[1].sub1 = 62.2;
    student[1].sub2 = 75.1;
    student[1].sub3 = 84.2;

    student[2].id = 21000003;
    strcpy(student[2].name, "Alice");
    student[2].sub1 = 82.4;
    student[2].sub2 = 65.4;
    student[2].sub3 = 91.1;

    student[3].id = 21000004;
    strcpy(student[3].name, "Bob");
    student[3].sub1 = 78.1;
```

Name: Dulapah Vibulsanti

ID: 64011388

```
student[3].sub2 = 90.3;
student[3].sub3 = 71.5;

student[4].id = 21000005;
strcpy(student[4].name, "Mary");
student[4].sub1 = 90.3;
student[4].sub2 = 82.5;
student[4].sub3 = 61.3;

float total[3] = {0};
for (int i = 0; i < 5; i++) {
    total[0] += student[i].sub1 / 5;
    total[1] += student[i].sub2 / 5;
    total[2] += student[i].sub3 / 5;
}
for (int i = 0; i < 3; i++)
    printf("Sub%d average = %.1f\n", i, total[i]);
return 0;
}
```

```
Sub0 average = 77.1
Sub1 average = 79.7
Sub2 average = 73.5
```

3.2) From 3.1) write a program to calculate average score for all subjects for each person.

Score average for Joe = 72.37
Score average for John = 73.83
Score average for Alice = 79.63
Score average for Bob = 79.97
Score average for Mary = 78.03

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

struct Student {
    int id;
    char name[30];
    float sub1, sub2, sub3;
};

int main() {
    struct Student student[5];

    student[0].id = 21000001;
    strcpy(student[0].name, "Joe");
    student[0].sub1 = 72.5;
    student[0].sub2 = 85.2;
    student[0].sub3 = 59.4;

    student[1].id = 21000002;
    strcpy(student[1].name, "John");
    student[1].sub1 = 62.2;
    student[1].sub2 = 75.1;
    student[1].sub3 = 84.2;

    student[2].id = 21000003;
    strcpy(student[2].name, "Alice");
    student[2].sub1 = 82.4;
    student[2].sub2 = 65.4;
    student[2].sub3 = 91.1;

    student[3].id = 21000004;
    strcpy(student[3].name, "Bob");
    student[3].sub1 = 78.1;
    student[3].sub2 = 90.3;
    student[3].sub3 = 71.5;

    student[4].id = 21000005;
    strcpy(student[4].name, "Mary");
    student[4].sub1 = 90.3;
    student[4].sub2 = 82.5;
```

Name: Dulapah Vibulsanti

ID: 64011388

```
    student[4].sub3 = 61.3;

    float avg[5] = {0};
    for (int i = 0; i < 5; i++) {
        avg[i] += (student[i].sub1 + student[i].sub2 + student[i].sub3) / 3;
        printf("Score average for %s = %.2f\n", student[i].name, avg[i]);
    }
    return 0;
}
```

```
Score average for Joe = 72.37
Score average for John = 73.83
Score average for Alice = 79.63
Score average for Bob = 79.97
Score average for Mary = 78.03
```

3.3) Write a program to print the data to the screen in the following order:

21000003	Alice	82.4	65.4	91.1
21000005	Mary	90.3	82.5	61.3
21000002	John	62.2	75.1	84.2
21000001	Joe	72.5	85.2	59.4
21000004	Bob	78.1	90.3	71.5

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

struct Student {
    int id;
    char name[30];
    float sub1, sub2, sub3;
};

int main() {
    struct Student student[5];

    student[0].id = 21000001;
    strcpy(student[0].name, "Joe");
    student[0].sub1 = 72.5;
    student[0].sub2 = 85.2;
    student[0].sub3 = 59.4;

    student[1].id = 21000002;
    strcpy(student[1].name, "John");
    student[1].sub1 = 62.2;
    student[1].sub2 = 75.1;
    student[1].sub3 = 84.2;

    student[2].id = 21000003;
    strcpy(student[2].name, "Alice");
    student[2].sub1 = 82.4;
    student[2].sub2 = 65.4;
    student[2].sub3 = 91.1;

    student[3].id = 21000004;
    strcpy(student[3].name, "Bob");
    student[3].sub1 = 78.1;
    student[3].sub2 = 90.3;
    student[3].sub3 = 71.5;

    student[4].id = 21000005;
    strcpy(student[4].name, "Mary");
    student[4].sub1 = 90.3;
```


Name: Dulapah Vibulsanti

ID: 64011388

```
    student[4].sub2 = 82.5;
    student[4].sub3 = 61.3;

    int arr[] = {2, 4, 1, 0, 3};
    for (int i = 0; i < 5; i++)
        printf("%d %s\t%.1f %.1f %.1f\n", student[arr[i]].id, student[arr[i]].name, student[arr[i]].sub1, student[arr[i]].sub2, student[arr[i]].sub3);
    return 0;
}
```

```
21000003 Alice 82.4 65.4 91.1
21000005 Mary 90.3 82.5 61.3
21000002 John 62.2 75.1 84.2
21000001 Joe 72.5 85.2 59.4
21000004 Bob 78.1 90.3 71.5
```