2023 年度 ソフトウェア構成論
レポート 1

学修番号: 22140003
氏名: 佐倉仙汰郎

第 1 回レポート提出日：2023/12/22

1 問題１

問題 1：円の面積を求める関数をマクロ置換により作成し，半径 30 の円の面積を求めなさい．ただし，円周率は C 言語の標準ライブラリより読み込むこと．

```
    #include<iostream>
#include<cmath>

using namespace std;

#define PI 3.1414
#define area(r) (M_PI * r * r)

int main(){
    double r = 30.0;
    double area = area(r);

    cout << area <<endl;
    return 0;
}
```

出力結果

2827.43

2 問題２

```
    #include<iostream>
#include<cmath>

using namespace std;

void calc(double *xn){
    *xn = sqrt(*xn) + 1;
}

int main(){
    double x0;
    for(int i =0; i < 20; i++){
        calc(&x0);
        cout << x0 << endl;
    }
```

```
    return 0;
}
```

出力結果

```
1
2
2.41421
2.55377
2.59805
2.61185
2.61612
2.61744
2.61785
2.61798
2.61802
2.61803
2.61803
2.61803
2.61803
2.61803
2.61803
2.61803
2.61803
2.61803
```

3 問題３

```
    /*問題 3：stdlib.h の qsort 関数など，標準ライブラリに含まれているソート関数では，
2 つの変数を比較するための関数を定義することにより，並べ替えの基準を自由に設定
することができる．独自の比較関数を定義し，並べ替えを実行した例を示しなさい．並
べ替える変数は構造体など何でもよい．*/

#include <stdlib.h>
#include<iostream>

using namespace std;

int comp( const void* lhs, const void* rhs ) {
    return *(char*)lhs - *(char*)rhs;
}

typedef struct {
```

```
    char s;
} test;

int main() {
    // 構造体を要素とする配列を作成
    test array[] = {
        {'e'},
        {'a'},
        {'k'},
        {'y'},
        {'u'}
    };

    size_t array_size = sizeof(array) / sizeof(array[0]);

    qsort(array, array_size, sizeof(test), comp);

    for (size_t i = 0; i < array_size; ++i) {
        printf("%c ", array[i].s);
    }
    return 0;
}
```

出力結果

a e k u y

4 問題４

```
    #include<iostream>

using namespace std;

void plus5_with_pointer(int* x){
    *x = *x + 5;
}

void plus5_without_pointer(int x){
    x = x + 5;
}

int main(){
    int x = 5;
```

```
    int *p = &x;

    cout << "ポインタを使ってアドレスを使う方法はいくつかある．以下では２つのやり方を示す．" <<
        endl;
    cout << "xのアドレス: "<< p << endl;
    cout << "xのアドレス: "<< &x << endl;
    cout << "値を表す方法も示す．" << endl;
    cout << "xの値: " << *p << endl;

    cout << "xの値を関数を用いて変更したいときのもポインタが使える．" << endl;
    plus5_without_pointer(x);
    cout << "ポインタを使わない関数で＋５した場合、x: " << x << endl;
    plus5_with_pointer(p);
    cout << "ポインタを使った関数で＋５したとき、x: " << x << endl;

    cout << "----------------------------------------" << endl;
    cout << "ポインタは配列についても有用である．" << endl;
    int vals[] = { 1 ,1 ,2 ,3 ,5, 8, 13};
    int *valptr = vals;
    cout << "配列の先頭の値: "<<*valptr << endl;
    for(int i = 0; i < 7; i++){
        cout << *(valptr + i) << " ";
    }
    return 0;



    return 0;
}
```

出力結果

```
ポインタを使ってアドレスを使う方法はいくつかある．以下では２つのやり方を示す．
x のアドレス: 0x7ffd3bb15878
x のアドレス: 0x7ffd3bb15878
値を表す方法も示す．
x の値: 5
x の値を関数を用いて変更したいときのもポインタが使える．
ポインタを使わない関数で＋５した場合、x: 5
ポインタを使った関数で＋５したとき、x: 10
----------------------------------------
ポインタは配列についても有用である．
配列の先頭の値: 1
1 1 2 3 5 8 13
```